

SunLink™ Client 3270 9.1 Configuration and User's Manual



The Network Is the Computer™

Sun Microsystems Computer Company

2550 Garcia Avenue
Mountain View, CA 94043 USA
415 960-1300 fax 415 969-9131

Part No.: 802-2667-12
Revision A, August 1997

Copyright 1997 Sun Microsystems, Inc. 2550 Garcia Avenue, Mountain View, California 94043-1100 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of this product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U. S. and other countries, exclusively licensed through X/Open Company Ltd.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g)(2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015(b)(6/95) and DFAR 227.7202-3(a).

Sun, Sun Microsystems, the Sun logo, AnswerBook, SunDocs, SunLink, OpenWindows, and Solaris are trademarks, registered trademarks, or service marks of Sun Microsystems, Inc. in the U. S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U. S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 1997 Sun Microsystems, Inc., 2550 Garcia Avenue, Mountain View, Californie 94043-1100 U.S.A. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie et la décompilation. Aucune partie de ce produit ou de sa documentation associée ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.

Des parties de ce produit pourront être dérivées du système UNIX® licencié par Novell, Inc. et du système Berkeley 4.3 BSD licencié par l'Université de Californie. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays, et licenciée exclusivement par X/Open Company Ltd. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Sun, Sun Microsystems, le logo Sun, AnswerBook, SunDocs, SunLink, OpenWindows, et Solaris sont des marques déposées ou enregistrées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC, utilisées sous licence, sont des marques déposées ou enregistrées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

Les utilisateurs d'interfaces graphiques OPEN LOOK® et Sun™ ont été développés de Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox Corporation pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique, cette licence couvrant aussi les licenciés de Sun qui mettent en place les utilisateurs d'interfaces graphiques OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" SANS GARANTIE D'AUCUNE SORTE, NI EXPRESSE NI IMPLICITE, Y COMPRIS, ET SANS QUE CETTE LISTE NE SOIT LIMITATIVE, DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DES PRODUITS A REpondre A UNE UTILISATION PARTICULIERE OU LE FAIT QU'ILS NE SOIENT PAS CONTREFAISANTS DE PRODUITS DE TIERS.



Contents

Preface	xv
1. Introduction to SUNWopcl	1-1
1.1 SUNWopcl Client	1-3
1.2 SUNWopcl Emulation	1-4
1.3 Keyboard Mapping	1-7
1.4 Configuring SUNWopcl	1-7
2. Getting Started with SUNWopcl	2-1
2.1 Installing SUNWopcl	2-2
2.2 Coordinating Configurations	2-3
2.3 Configuring SUNWopcl	2-3
2.4 Starting sunpu2.1	2-4
2.5 Starting SUNWopcl	2-5
2.6 Stopping sunpu2.1	2-6
3. Using sun3270x	3-1
3.1 Starting sun3270x	3-2

3.1.1	Dependencies.....	3-2
3.1.2	Keywords.....	3-3
3.1.3	LU Attachment Keywords.....	3-3
3.1.4	Keyboard Map Attributes	3-4
3.1.5	EHLAPI Session Name	3-5
3.1.6	Color Display Attributes	3-6
3.1.7	Displaying Window Color Functions.....	3-7
3.1.8	Main Display Window Attributes.....	3-9
3.1.9	X Keywords	3-10
3.1.10	Miscellaneous Functions	3-11
3.1.11	File Transfer Menu Defaults	3-15
3.1.12	Light Pen Defaults	3-18
3.1.13	Host Graphics Keywords.....	3-19
3.1.14	User Button and Function Keywords.....	3-21
3.1.15	sun3270x Command Line Configuration Examples	3-22
3.2	sun3270x Window Layout.....	3-23
3.2.1	File Menu	3-25
3.2.2	Edit Menu.....	3-25
3.2.3	Action Menu	3-27
3.2.4	Settings Menu	3-27
3.3	Stopping sun3270x.....	3-28
3.4	Status Line.....	3-28
3.5	Data Entry Overview	3-30
3.5.1	SunLink 3270 Screen Formats	3-30

3.5.2	3270 Keyboard Input Summary	3-30
3.5.3	Keyboard Functions	3-31
3.6	Data Entry	3-33
3.7	Cursor Control	3-34
3.7.1	Moving the Cursor from Character to Character ..	3-34
3.7.2	Moving the Cursor from Field to Field	3-35
3.8	Start/Stop Data Entry	3-36
3.9	Editing	3-38
3.10	Field Control	3-39
3.11	Special Functions	3-40
4.	Customizing sun3270x	4-1
4.1	sun3270x Resources	4-1
4.2	Example SUNWopc1 Resource File	4-3
4.3	Settings: Dynamic SUNWopc1 Resource Updates	4-4
4.4	Message Library Updates	4-8
4.5	ASCII/EBCDIC Translations	4-9
4.5.1	Sample ASCII/EBCDIC Translation Table	4-10
5.	Using Keyboard Mapper	5-1
5.1	Displaying the Keyboard Map	5-1
5.2	Keyboard Map Layout	5-2
5.2.1	Menu Bar	5-4
5.2.2	Title Bar	5-4
5.2.3	Unmapped IBM Keys	5-5
5.2.4	Mode Button	5-5

5.2.5	Single Key Layout.....	5-5
5.2.6	Keyboard Layout.....	5-6
5.3	Updating the Keyboard Map	5-6
5.3.1	Moving an IBM Key	5-6
5.3.2	Duplicating an IBM Key.....	5-8
5.3.3	Removing an IBM Key	5-9
5.4	Saving Keyboard Map Updates	5-9
5.5	Exiting Keyboard Map Display	5-9
6.	Using File Transfer	6-1
6.1	File Transfer Dependencies	6-1
6.2	Starting a File Transfer.....	6-2
6.3	File Transfer Status.....	6-4
6.4	File Transfer Options	6-4
7.	Using sun3287	7-1
7.1	Dependencies	7-2
7.2	Starting sun3287	7-2
7.3	Print Stream File Manipulation Examples.....	7-7
7.4	Starting suntn3287.....	7-7
8.	SunLink PU2.1 SNA	
	Server Configuration	8-1
8.1	SunLink PU2.1 SNA Server Configuration	8-1
8.2	LU Directive	8-2
8.3	Example Display Terminal Configuration.....	8-4
8.4	Sample SNA Server Configuration.....	8-4

9. SNA Configuration for SUNWopcl.....	9-1
9.1 SNA Configuration.....	9-1
9.2 LOGMODE Tables	9-5
9.3 Application Tables	9-7
10. SUNWopcl Troubleshooting.....	10-1
10.1 Online Help.....	10-2
10.2 Operator Status Information.....	10-2
10.3 Logical Data Scope and Tracing	10-3
10.4 Recovery Steps	10-4
10.5 Common Problems.....	10-5
11. Using suntn3270x.....	11-1
11.1 suntn3270 Dependencies	11-2
11.2 suntn3270 Keywords.....	11-2
11.2.1 Miscellaneous Keywords.....	11-3
11.2.2 Host Graphics Keywords.....	11-4
11.3 Stopping suntn3270x.....	11-6
11.4 suntn3270 Status Line.....	11-6
11.5 suntn3270 Examples	11-7
12. Using CG3270	12-1
12.1 Additional information.....	12-3
13. Using sun3270tty.....	13-1
13.1 sun3270tty Dependencies	13-2
13.1.1 sun3270tty Status Line	13-2
13.1.2 Intensified Support	13-3

13.1.3	sun3270tty Keywords.....	13-3
13.1.4	sun3270tty Examples	13-6
A.	Customizing the sun3270x Keyboard	A-1
A.1	Starting sunke.....	A-1
A.1.1	Files.....	A-2
A.1.2	Dependencies.....	A-2
A.2	Keywords.....	A-3
A.2.1	Keyboard Layout Definition Files.....	A-3
A.2.2	Window Attributes	A-4
A.2.3	Window Colors	A-6
A.2.4	Generic Arguments.....	A-8
A.3	Customizing sunke	A-8
A.4	Creating a New Keyboard Layout	A-9
A.4.1	Defining an IBM Key Caps File	A-10
A.5	Sample IBM Key Caps File	A-13
A.6	Defining a Keyboard Geometry File.....	A-19
A.6.1	Sample Keyboard Geometry File	A-20
A.7	Teaching sunke Keyboard Functions	A-28
A.7.1	Key Characterization	A-28
A.8	Mapping IBM Key Caps	A-31
B.	Mapping sun3270tty Keyboards.....	B-1
B.1	Keyboard Map File Format	B-2
B.1.1	ibmCaps Record	B-3
B.1.2	aliases Record	B-5

B.1.3	Terminal Map Record	B-5
B.1.4	ASCII Character Sequence	B-6
B.1.5	Sample sun3270map File	B-7
C.	Using PCFT	C-1
C.1	TSO Options	C-3
C.1.1	Send (TSO)	C-3
C.1.2	Send (TSO) Examples	C-5
C.1.3	Receive (TSO)	C-6
C.1.4	Receive (TSO) Examples	C-7
C.2	CICS Options	C-7
C.2.1	Send (CICS)	C-8
C.2.2	Send (CICS) Examples:	C-9
C.2.3	Receive (CICS)	C-9
C.2.4	Receive (CICS) Examples	C-10
C.3	VM / CMS Options	C-10
C.3.1	Send (VM / CMS)	C-11
C.3.2	Send (VM / CMS) Examples	C-12
C.4	Receive (VM / CMS)	C-12
C.4.1	Receive (VM / CMS) Examples	C-13
D.	SunLink 3270 Tracing	D-1
E.	Error Messages	E-1
E.1	bmsg: Sun Basic Message Display Utility	E-2
E.2	BMD: Sun Basic Message Database	E-2
E.3	Message Files	E-3

F. DBCS.....	F-1
F.1 DBCS Keywords.....	F-1
Index	Index-1

Figures

Figure 1-1	SUNWopcl and the sunpu2.1 SNA Server	1-2
Figure 1-2	SUNWopcl and SNA	1-3
Figure 1-3	SUNWopcl Screen	1-6
Figure 2-1	Sample Configuration.....	2-2
Figure 3-1	The sun3270x Window	3-24
Figure 3-2	Sample Status Line	3-28
Figure 3-3	Light Pen - Cut/Copy - Paste Menu.....	3-41
Figure 4-1	Options and Settings.....	4-5
Figure 5-1	Options Keyboard Mapper	5-1
Figure 5-2	sun3270x Keyboard Map Layout	5-3
Figure 5-3	Single Key Format	5-6
Figure 5-4	Previously Unmapped IBM Key Value (F20) Being Mapped	5-7
Figure 5-5	IBM Key Value (Btab) Being Re-mapped.....	5-8
Figure 5-6	File Menu	5-9
Figure 6-1	Action to File Transfer	6-2
Figure 6-2	File Transfer	6-3

Figure 8-1	Point-to-Point Line Configuration Workstation	8-5
Figure 9-1	Sample SNA Configuration	9-2
Figure 11-1	suntn3270 Status Line	11-6
Figure A-1	Key Characterization Menu	A-29
Figure A-2	sun3270x Print Key Characterization.	A-30
Figure A-3	IBM RS-6000 Print Key Characterization	A-31

Tables

Table 1-1	Applications and their Emulation.....	1-5
Table 2-1	Example Configuration Parameters	2-4
Table 3-1	IBM Color Keywords and Display Attributes.....	3-6
Table 3-2	Status Line Meaning	3-29
Table 3-3	Keyboard Functions	3-32
Table 5-1	Keyboard Mapper Modes.....	5-5
Table 6-1	File Transfer Options.....	6-4
Table 7-1	SCS Control Codes	7-1
Table 7-2	Horizontal Format Specifiers.....	7-4
Table 7-3	Vertical Format Specifiers	7-5
Table 8-1	Configuration Syntax	8-2
Table 8-2	LU Port Attributes.....	8-2
Table 8-3	LU Types and Sun Client Programs.....	8-3
Table 8-4	SNA 3270 Display Configuration Example.....	8-4
Table 8-5	IBM SNA 3287 Printer Configuration Example	8-4
Table 8-6	Server Configuration Parameters	8-6

Table 9-1	NCP / VTAM Macros.....	9-2
Table 10-1	Common Problems and Solutions	10-5
Table 11-1	suntn3270 Status Line	11-7
Table A-1	Keyboard Functions	A-11
Table B-1	sun3270tty Keyboard Functions.....	B-4
Table C-1	pcft Command Options	C-2
Table C-2	Send (TSO) Syntax.....	C-4
Table C-3	Receive (TSO) Syntax	C-6
Table C-4	Send (CICS) Syntax	C-8
Table C-5	Receive (CICS) Syntax.....	C-9
Table C-6	Send (VM / CMS) Syntax.....	C-11
Table C-7	Receive (VM / CMS) Syntax	C-13
Table E-1	Message String Components	E-4

Preface

This manual describes the administration and use of the Sun SNA 3270 emulation product, **SUNWopc1**, which is a client application of the SunLink™ SNA 9.1 PU2.1 server.

Who Should Use This Book

The *SunLink Client 3270 9.1 Configuration and User's Guide* is the user's guide for the **SUNWopc1** product. It describes how to install and configure the **SUNWopc1** product. It provides information on how to invoke **SUNWopc1** capabilities as well as to map the behavior of real 3270 devices to the same functions that are provided by the **SUNWopc1** product. Extended High-Level Language Application Program Interface (EHLAPI) programmers need to refer to this manual to correctly invoke **SUNWopc1** as an EHLAPI server application.

System programmers define the **SUNWopc1** devices in the SNA configuration, add these new devices to the SunLink **sunpu2.1** SNA server configuration, and customize the **SUNWopc1** client software. Usually, one system programmer is responsible for the SNA network updates, another installs the **sunpu2.1** server and builds the local configuration file, and possibly a third installs the **SUNWopc1** client software. These system programmers must coordinate the parameters chosen for the SNA configuration, the **sunpu2.1** server and the **SUNWopc1** configurations.

As a system programmer responsible for the SNA network configuration, you should be familiar with adding new devices to a Virtual Terminal Access Method/Network Control Point Generation (VTAM/NCP GEN).

As a system programmer responsible for installing and configuring the SunLink sunpu2.1 SNA server, you should be familiar with the Solaris™ operating system and the configuration of SNA devices.

As a system programmer or administrator responsible for the SUNWopcl client installation and configuration, you should be familiar with the Solaris operating system and its configuration.

As a user, you should understand the 3270 data entry and display characteristics.

As a system operator, you will learn how to invoke the sun3270x emulation capabilities as well as how to map the behavior of real 3270 devices to the same functions provided by sun3270x. You should understand how SUNWopcl interacts with the SNA network, as described in Chapter 1.

If you are an EHLLAPPI programmer, this manual will show you how to invoke SUNWopcl as an EHLLAPI server application.

For Pacific Rim customers, this manual includes information on DBCS functionality. Appendix D, “SunLink 3270 Tracing,” contains additional keywords.

How This Book Is Organized

This book is organized as follows:

Chapter 1, “Introduction to SUNWopcl,” provides a conceptual overview of SUNWopcl and serves as background for subsequent chapters.

Chapter 2, “Getting Started with SUNWopcl,” describes the steps you take to install, configure, and start SUNWopcl.

Chapter 3, “Using sun3270x,” specifies how to use the SUNWopcl data entry and display characteristics in the Solaris environment.

Chapter 4, “Customizing sun3270x,” defines how to change the various attributes of the SUNWopcl display via the SUNWopcl pull-down menus.

Chapter 5, “Using Keyboard Mapper,” describes the SUNWopcl graphical keyboard map, which displays the mapping of SUNWopcl keyboard functions to keys available on the local Sun workstation terminal. System programmers should provide their users with a generic mapping. Operators should refer to this section if they want to change their keyboard mappings for SUNWopcl.

Chapter 6, “Using File Transfer,” describes how to exchange files between the local system and the IBM host using the SUNWopcl menu-driven file transfer mechanism.

Chapter 7, “Using sun3287,” specifies the startup requirements and printer output functions for the sun3287 printer emulation program.

Chapter 8, “SunLink PU2.1 SNA Server Configuration,” specifies the syntax and descriptions of the SUNWopcl configuration directives in the sunpu2.1 SNA server configuration.

Chapter 9, “SNA Configuration for SUNWopcl,” describes the updates required for the SNA host network configuration.

Chapter 10, “SUNWopcl Troubleshooting,” provides troubleshooting procedures for SUNWopcl.

Chapter 11, “Using suntn3270x,” describes how to start and run the Sun X-based Telnet 3270 emulator (suntn3270x). The suntn3270x program provides Telnet 3270 emulation for any system supporting the X Window™ standard.

Chapter 12, “Using CG3270,” explains how to use CG3270 emulation, which adds the capability to support GDDM commands from the IBM host system.

Chapter 13, “Using sun3270tty,” describes how to start and run Sun's TTY-based SNA 3270 emulator, sun3270tty. The sunw3270tty program is a Unix application. Many sun3270tty programs can run in parallel. Each sun3270tty controls one terminal and one session with an IBM host application.

Appendix A, “Customizing the sun3270x Keyboard,” describes how to customize your keyboard using the sunke Unix mapper program.

Appendix B, “Mapping sun3270tty Keyboards,” describes how to map the sun3270tty keyboards.

Appendix C, “Using PCFT,” describes how to use the PCFT program. SUNWopc1 supports the transfer of binary, ASCII, and EBCDIC text files between the local Unix system and IBM host systems that support the IBM 3270 PC file transfer option.

Appendix D, “SunLink 3270 Tracing,” explains how to turn on the trace utility and start the appropriate emulator with the -t option. Trace points record all information received and sent to the sunpu2.1 SNA server, and all information received and sent to the terminal. It explains how to turn on the trace utility and start the appropriate emulator with the -t option.

Appendix E, “Error Messages,” describes the error messages generated by the system. All informational and error messages generated by Sun software are derived from the BMD. It describes how to use the bmsg utility.

Appendix F, “DBCS,” describes the double-byte character string for Pacific Rim users.

Related Documents

The following Sun documents contain topics that relate to the information in *SunLink Client 3270 9.1 Configuration and User's Guide*.

Purpose	Document Title	Part Number
Installation	<i>SunLink SNA 9.1 End Node Planning and Installation Guide</i>	802-2665-12
Administration	<i>SunLink SNA/X.25 9.1 Configuration and Administration Guide</i>	802-3166-12
Configuration and user information	<i>SunLink 3270 OpenClient 9.1 Configuration and User's Guide</i>	802-2667-12
Configuration and user information	<i>SunLink RJE/3770 9.1 Configuration and User's Guide</i>	802-2672-12
Configuration information	<i>SunLink SNA PU 2.1 9.1 Server Configuration Guide</i>	802-2673-12
Programming information	<i>SunLink Client IBM 3270 9.1 Programmer's Guide</i>	802-2668-12
Programming information	<i>SunLink LU0 9.1 API Programmer's Guide</i>	802-2676-12
Programming information	<i>SunLink SNA Peer to Peer LU 6.2 9.1 Programmer's Guide</i>	802-2680-12
Programming information	<i>SunLink SNA Peer to Peer CPI-C 9.1 Programmer's Guide</i>	802-2681-12
Reference information	<i>SunLink SNA SNM 9.1 Reference Manual</i>	802-2674-12
Late-breaking news	<i>SunLink SNA 9.1 for Solaris Release Notes</i>	802-3165-12

IBM Documentation

The following IBM documents provide reference information.

- *IBM Systems Network Architecture Concepts and Products (GC30-3072)*
- *IBM VTAM Installation and Resource Definition (SC23-0111)*
- *IBM Network Control Program Resource Definition Guide (SC30-3349)*
- *IBM VTAM Operation (SC23-0113)*
- *IBM NetView Operation (SC30-3364)*

Typographic Conventions

The following table describes the typographic changes used in this book.

Typeface or Symbol	Meaning	Example
AaBbCc123	The names of commands, files, and directories; on-screen computer output.	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. % You have mail.
AaBbCc123	What you type, when contrasted with on-screen computer output.	% su Password:
<i>AaBbCc123</i>	Command-line variable: replace with a real name or value.	To delete a file, type <code>rm filename</code> .
	Book titles, new words or terms, words to be emphasized	Read Chapter 6 in the <i>User's Guide</i> . These are called <i>class</i> options. You <i>must</i> be root to do this.

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

Table P-1 Shell Prompts

Shell	Prompt
C shell	machine_name%
C shell superuser	machine_name#
Bourne shell and Korn shell	\$
Bourne shell and Korn shell superuser	#

Ordering Sun Documents

SunDocsSM is a distribution program for Sun Microsystems technical documentation. Easy, convenient ordering and quick delivery is available from SunExpress. You can find a full listing of available documentation on the World Wide Web: <http://www.sun.com/sunexpress/>

Table P-2 SunExpress Contact Information

Country	Telephone	Fax
United States	1-800-873-7869	1-800-944-0661
United Kingdom	0800-89-88-88	0800-89-88-87
Canada	1-800-873-7869	1-800-944-0661
France	0800-90-61-57	0800-90-61-58
Belgium	02-720-09-09	02-725-88-50
Luxembourg	32-2-720-09-09	32-2-725-88-50
Germany	01-30-81-61-91	01-30-81-61-92
The Netherlands	06-022-34-45	06-022-34-46
Sweden	020-79-57-26	020-79-57-27

Table P-2 SunExpress Contact Information (Continued)

Switzerland	0800-55-19-26	0800-55-19-27
Holland	06-022-34-45	06-022-34-46
Japan	0120-33-9096	0120-33-9097

Sun Welcomes Your Comments

Please use the *Reader Comment Card* that accompanies this document. We are interested in improving our documentation and welcome your comments and suggestions.

If a card is not available, you can email or fax your comments to us. Please include the part number of your document in the subject line of your email or fax message.

- Email: smcc-docs@sun.com
- Fax: SMCC Document Feedback
1-415-786-6443

Introduction to SUNWopc1



The SUNWopc1 software emulates IBM SNA 3270 display terminals on Sun computers, enabling users to interactively access IBM mainframe applications such as CICS, IMS, TSO, and NetView from a workstation (using a graphical user interface) or from a serial-attached TTY terminal.

The SUNWopc1 product supports two display interfaces:

- OpenWindows™ for sun3270x on systems supporting the Sun OpenWindows interface
- TTY-type terminals for sun3270tty on terminals like VT100s, VT200s, and Wyse

The SUNWopc1 product has the following capabilities:

- 3278 model 1, 2, 3, 4, and 5 emulation
- 3279 model 2A, 2B, 3A, and 3B emulation
- Extended High-Level Language Application Program Interface (EHLLAPI) program services
- File transfer (IND\$FILE) support
- 3287 printer emulation (LU Type 1 and LU Type 3)
- Graphical keyboard mapping (keyboard layout display, drag and drop keys to remap)

In addition, SUNWopc1, which includes the 3270 data stream:

- Controls the local screen display
- Maps keyboard functions
- Converts ASCII to EBCDIC, and vice versa

The SUNWopc1 is responsible for presenting the 3270 data stream. Therefore, the controls on the local screen display maps keyboard functions, and converts ASCII to EBCDIC, and vice versa. It uses the sunpu2.1 SNA server to establish and control LU Type 1, LU Type 2, and LU Type 3 SNA sessions with mainframe applications.

The physical connection to the SNA network is provided by the sunpu2.1 SNA server. The sunpu2.1 SNA server location is independent from where the SUNWopc1 client emulation actually occurs. The sunpu2.1 SNA gateway and the SUNWopc1 client can either execute on the same system, or can be separated and communicate over a TCP/IP network.

In Figure 1-1, the users of workstations on the TCP/IP LAN can access the mainframe by invoking SUNWopc1 in an open window, and then logging on to the mainframe applications.

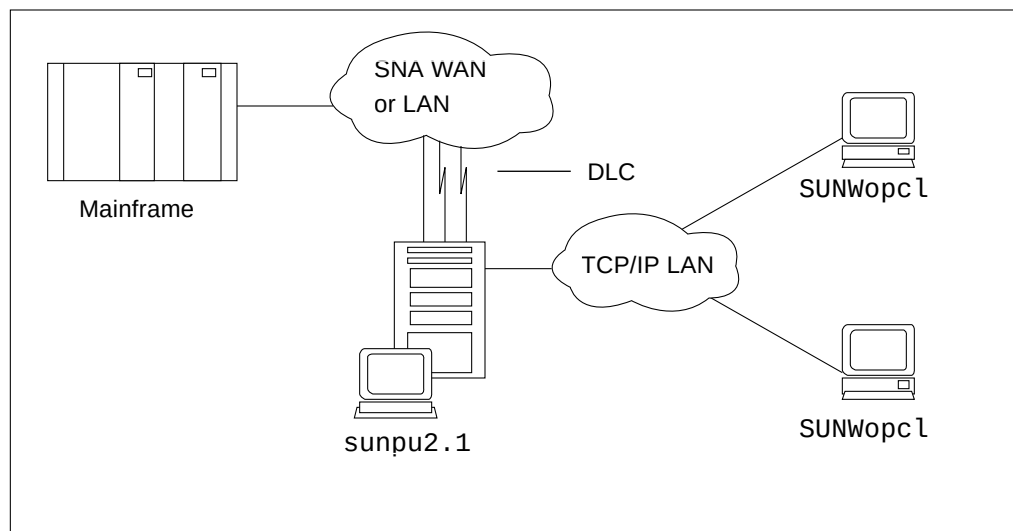


Figure 1-1 SUNWopc1 and the sunpu2.1 SNA Server

1.1 SUNWopcl Client

SUNWopcl has several different applications; each application provides different emulation capabilities (OpenWindows/Motif 3270 emulation, TTY 3270 emulation, and 3287 printer emulation). Each application, however, runs as an SNA client program on a Sun Workstation. They attach to a sunpu2.1 SNA server to access the SNA network. The relationship between the Sun SNA resources and their comparable IBM devices is illustrated in Figure 1-2.

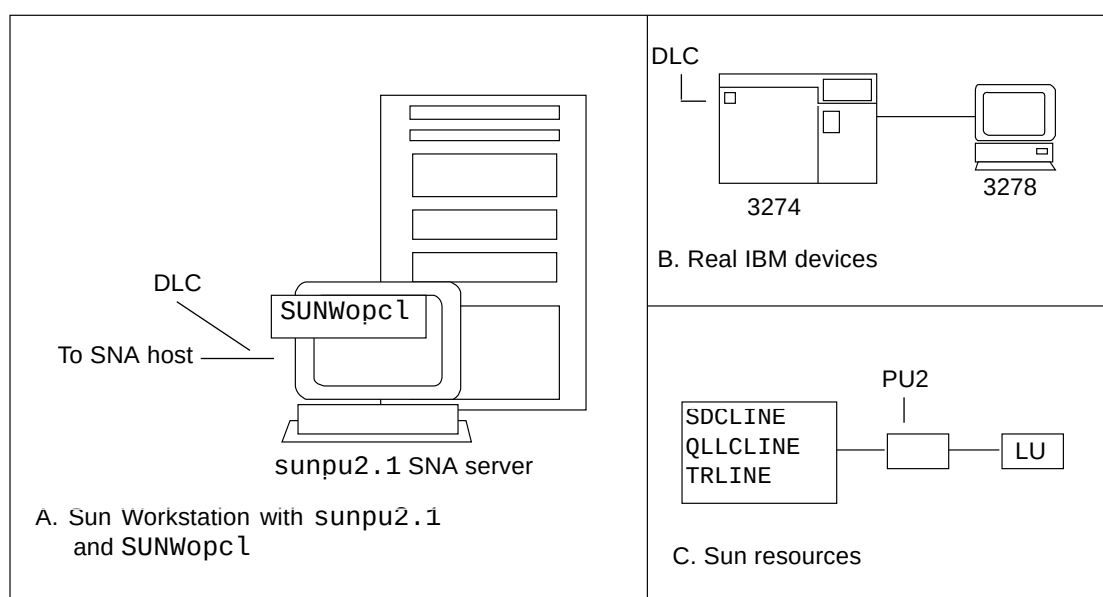


Figure 1-2 SUNWopcl and SNA

The SUNWopcl applications interpret the SNA 3270 data stream to provide presentation services. The sunpu2.1 SNA server emulates the functions of an SNA control unit such as an IBM 3174 cluster controller.

The relationship of the Sun SNA resources and the IBM devices, as illustrated in Figure 1-2, is as follows:

- Part A of Figure 1-2 illustrates the sunpu2.1 SNA server running on a Sun Workstation; a client SNA program, SUNWopcl, is also running on the Sun Workstation.

- Part B of Figure 1-2 shows a parallel configuration of IBM devices.
- Part C of Figure 1-2 relates the Sun SNA resources to the IBM devices, where the `sunpu2.1` SNA server represents the PU2 and attaches to the SNA network via an `SDLCLINE` (serial attachment), `TRLIN` (Token Ring attachment), or `QLLCLINE` (public data network attachment). `SUNWopcl` represents the LU.

1.2 `SUNWopcl` Emulation

`SUNWopcl` emulates IBM 3278 display terminals (models 1-5), IBM 3279 display terminals (models 2A, 2B, 3A, 3B), and IBM 3287 printers. The `SUNWopcl` applications are described in Table 1-1.

Table 1-1 Applications and their Emulation

Application	Emulation
sun3270x	OpenWindows / Motif 3270 emulation: IBM 3278 models 1-5 IBM 3279 models 2A, 2B, 3A, 3B EHLLAPI program services File transfer Resizable display window Keyboard mapping TN3270 RFC 1646/1647 support
sun3270tty	TTY 3270 emulation: IBM 3278 models 1-5 EHLLAPI program services File transfer Keyboard mapping TN3270 RFC 1646/1647 support
sun3287	3287 printer emulation: LU Type 1 LU Type 3 Output print data to file, printer, or program

To provide SNA 3270 emulation, the **SUNWopcl** application controls the screen format and the display of characters. The 3270 data stream is converted to the appropriate screen commands and EBCDIC characters are mapped to corresponding ASCII characters. Likewise, the user input is displayed according to the 3270 display formats, and data is sent to the host in 3270 data stream-compliant format. A sample **SUNWopcl** screen is illustrated in Figure 1-3.

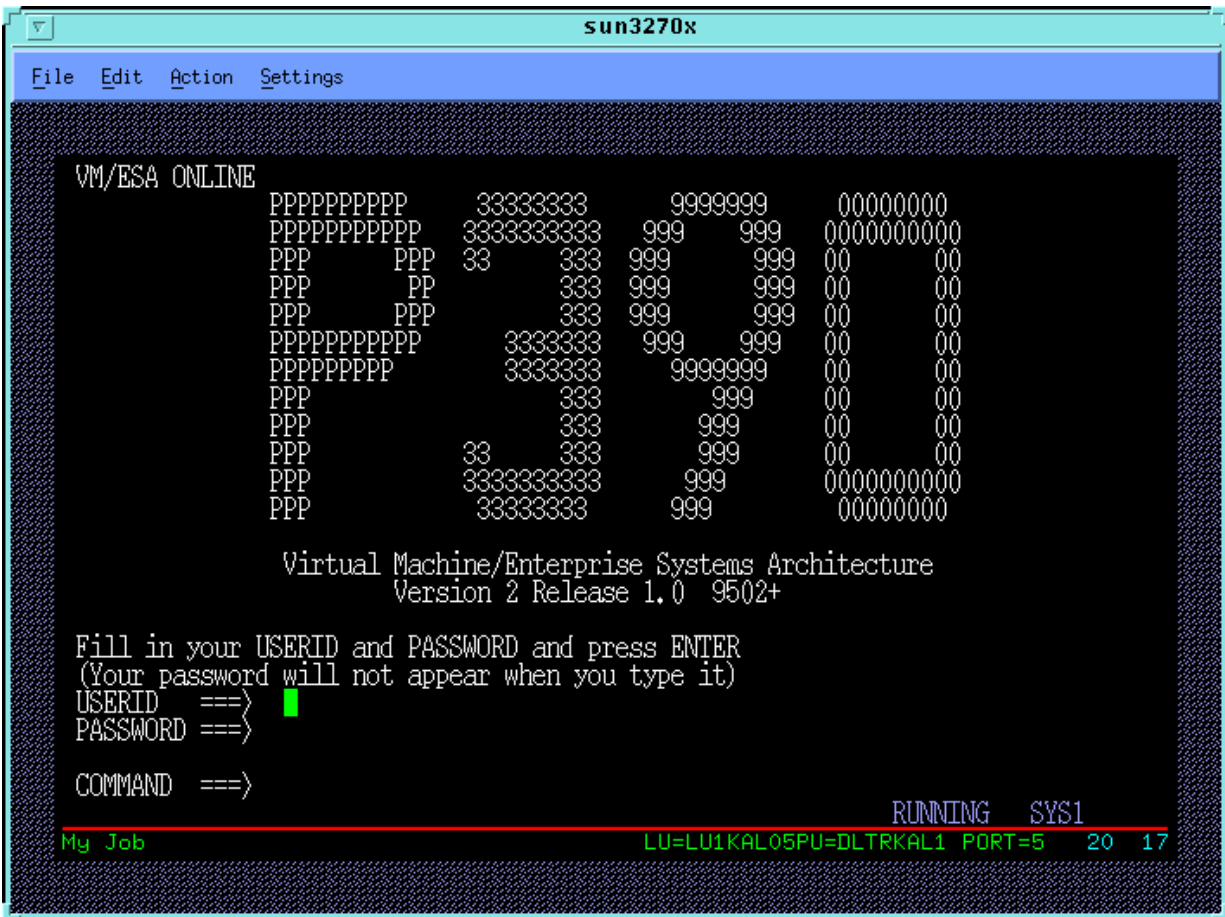


Figure 1-3 SUNWopc1 Screen

SUNWopcl does not support the following IBM 3270 display terminal functions:

- Case or test key
- Keyclick
- Alternate cursor (GDDM support only)
- Double cursor movement
- Programmed symbols
- Data entry assist

1.3 Keyboard Mapping

Your keyboard and the IBM 3270 display station keyboard are different. Sun provides a graphical keyboard map utility with sun3270x to help you quickly determine how the IBM key values are associated with your keyboard functions. This graphical utility displays a layout of your keyboard with the keys labeled with their corresponding IBM key values. It also enables you to dynamically update the current keyboard map while you are using sun3270x. IBM key values are picked up from one key and placed on another.

Consequently, the keyboard map utility provides a comprehensive help facility and the freedom to remap your Sun keyboard values whenever you want.

Sun also provides a keyboard map utility for sun3270tty. This utility helps you quickly determine which keys affect the IBM keyboard functions. You can also use this utility to change the mapping between keystrokes and IBM keyboard functions.

1.4 Configuring SUNWopcl

The SUNWopcl program connects to LU Type 2 ports on the sunpu2.1 SNA server. The sun3287 printer emulator connects to LU Type 3 ports on the sunpu2.1 SNA server. You can either direct these emulation programs to connect to a specific port or to any available one.

LU ports and PU2s represent SNA resources. Therefore, they must be configured in the NCP GEN file using the LU and PU macros. Similar entries must be added to the sunpu2.1 SNA server configuration file for each LU port and PU2 device.

Note – One sunpu2.1 SNA server can support up to 254 LU ports.

Getting Started with SUNWopc1



This chapter is designed to get you started with SUNWopc1 using a sample configuration that comprises a single link connection into the SNA network, a single PU2, and a single SUNWopc1 program. Chapter 8, “SunLink PU2.1 SNA Server Configuration,” lists the sample configuration distributed with SUNWopc1. Once you are successfully connected to a remote application, you can extend your configuration to meet your specific requirements.

This chapter describes the steps necessary for SNA connectivity. It focuses on configuring and starting the sunpu2.1 SNA server, and starting SUNWopc1.

Before SUNWopc1 can communicate with IBM mainframe applications, you must correctly install, configure, and activate the sunpu2.1 SNA server. Sun delivers the sunpu2.1 SNA server with installation scripts and sample configurations to ensure that you can begin communicating with IBM mainframe applications immediately. The following steps are required to run SUNWopc1 and are covered in more detail in subsequent sections.

1. Install the SunLink SNA PU2.1 9.1 server software.
2. Coordinate local and target network configurations.
3. Update the sunpu2.1 SNA server local configuration.
4. Start the sunpu2.1 SNA server (sunpu2.1) and check the status.
5. Start sun3270.

Figure 2-1 illustrates one sample configuration that is distributed with **SUNWopc1** (see Chapter 8, “SunLink PU2.1 SNA Server Configuration”). The remaining sections use this sample configuration as an example. Several other example configurations are included with the **sunpu2.1** SNA server as well.

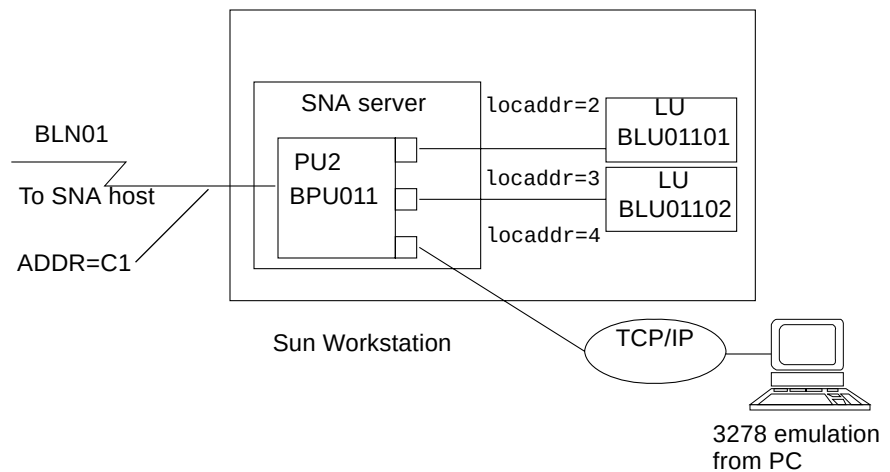


Figure 2-1 Sample Configuration

2.1 Installing SUNWopc1

Refer to the *SunLink SNA 9.1 End Node Planning and Installation Manual* for a complete description of the installation procedure. A summary of the installation process is as follows:

1. Install communications hardware and software, as necessary.
2. Remove and copy the SunLink **sunpu2.1** and **SUNWopc1** product files from the distribution media.
3. Install the SunLink software.

You can also run **SUNWopc1** on other systems in your network using the system running the **sunpu2.1** SNA server as a gateway to the SNA network. To install a remote **SUNWopc1**, move the **SUNWopc1** files to the remote system. The *SunLink SNA 9.1 End Node Planning and Installation Manual* discusses the

distribution of SUNWopcl programs in your network. When you start a remote SUNWopcl, you must specify the name of the system running the sunpu2.1 SNA server (use the `-h hostname` option).

2.2 Coordinating Configurations

The SNA resources described in the local sunpu2.1 SNA server configuration must mirror the values in the SNA host network configuration. For specific details about coordinating configurations, refer to the *SunLink SNA PU2.1 9.1 Server Configuration and Administration Manual*.

Consult your SNA host system programmer to determine the specific values for the parameters outlined below. After coordinating the updates to these two configurations, you are ready to establish a connection to the SNA network. Here are the parameters:

- Line attributes (SDLCLINE, TRLINE, or QLLCLINE parameters)
- Line station address (PU2 directive parameters)
- Port number of the logical unit (LOCADDR parameter of the LU directive)
- LU type (LUTYPE parameter for LU directive indicates display terminal or printer implied by LOGMODE parameters for the LU configured on the SNA host system)

Chapter 9, “SNA Configuration for SUNWopcl,” contains example SNA configuration information. This chapter contains parts of a VTAM/NCP GEN which illustrates the configuration of the SUNWopcl SNA resources (LINE, PU, and LU). This chapter also contains sample LOGMODE tables: one for SUNWopcl and one for sun3287 printer emulation.

2.3 Configuring SUNWopcl

To configure SUNWopcl SNA connections, use the configurations located in the Sun installation directory (and the ones listed in Chapter 8) or another sample configuration included in the *SunLink SNA PU2.1 9.1 Server Configuration and Administration Manual*.

Using the information in Section 2.2, “Coordinating Configurations,” update your configuration to reflect your local parameter values.

Note – A SUNWopcl display terminal emulator can only attach to LUs configured as `lutype=2` (the default). A sun3287 printer emulator can only attach to LUs configured as `lutype=3`. The sun3278 printer emulator will function as either a SNA LU Type 1 or LU Type 3, depending on the SNA BIND command received.

For the sample configuration, the particularly relevant parameters and their values in the `sunpu2.1` SNA configuration are listed in Table 2-1. Note that a 3287 printer is not configured.

Table 2-1 Example Configuration Parameters

sunpu2.1 Directive	sunpu2.1 Parameter	Parameter Value	Comment
SDLCLINE	DUPLEX	FULL	Point-to-point line
	LINE	LEASED	Line is non-switched
	NRZI	NO	Non-return to zero encoding
	SPEED	9600	Line speed in bits per second
PU2	ADDR	'C1'	Line station address
LU	LOCADDR	2	Logical port for LU
	LUTYPE	2	3270 display (default value)
LU	LOCADDR	3	Logical port for LU
	LUTYPE	2	3270 display (default value)
LU	LOCADDR	4	Logical port for LU
	LUTYPE	2	3270 display (default value)

2.4 Starting sunpu2.1

Note – The recommend method for starting `sunpu2.1` is to use the SunGMI.

After coordinating the SNA host network configuration and the local `sunpu2.1` SNA server configuration, you are ready to start the `sunpu2.1` SNA server. The *SunLink SNA PU2.1 9.1 Server Configuration and Administration Manual* lists the options for invoking the `sunpu2.1` process. This manual has a troubleshooting chapter (Chapter 10, “SUNWopcl Troubleshooting”) to aid in resolving SNA connectivity problems.

sunpu2.1 reads the local configuration file to learn about its SNA resources during initialization.

♦ **To display the status of the sunpu2.1 SNA server, use the sunop application, including LU status.**

The sunop application prompts you for management requests (-> is the sunop prompt).

♦ **To view the PU2 status, type dis (display status).**

```
-> dis
(2) dis
->
OP200025 : (2) Link BLN01 - (2) Active
OP200020 : (2)   Physical Unit BPU011 - (7) Active
OP20002a : (2)     Logical Unit BLU01101 - (2) Active
OP20002a : (2)     Logical Unit BLU01102 - (2) Active
OP20002a : (2)     Logical Unit BLU01103 - (2) Active
```

If the SNA host has successfully connected to the PU2, the Physical Unit status is Active. If the SNA host has successfully connected to the LU, the Logical Unit status is Active.

2.5 Starting SUNWopcl

After you successfully connect to the SNA network, you can start SUNWopcl.

♦ **To invoke SUNWopcl in the OpenWindows environment, type:**

```
% sun3270x
```

♦ **To invoke SUNWopcl in TTY-type terminal display, type:**

```
% sun3270tty
```

♦ **To start the SUNWopcl printer emulator, type:**

```
% sun3287
```

Chapter 3, “Using sun3270x” lists all the startup options for sun3270x. Appendix B lists all the startup options for sun3270tty. Chapter 7, “Using sun3287,” lists all the startup options for sun3287.

After starting SUNWopcl, your IBM mainframe banner should be displayed in the SUNWopcl window. If no banner is displayed, see Chapter 10, “SUNWopcl Troubleshooting.”

2.6 Stopping sunpu2.1

Note – The recommend method for stopping sunpu2.1 is to use the SunGMI.

To stop the sunpu2.1 SNA server:

1. **Load the sunop utility.**
2. **Type the kill command at the prompt.**

Using sun3270x



This chapter describes how to start and run the X/Motif-based SNA 3270 emulator (`sun3270x`). The `sun3270x` provides SNA 3270 emulation for any system that supports the OpenWindows environment.

The `SUNWopc1` product supports several graphical interfaces:

- The OpenWindows environment uses `sun3270x` on systems supporting OpenWindows, Open Look, Motif, XWindow Manager, workstations, PCs and xterms.
- TTY-type terminals use `sun3270tty` on terminals like VT100s, VT200s, and Wyse.

See Chapter 13, “Using `sun3270tty`” for more information about running `SUNWopc1` on TTY-type terminals.

The `sun3270x` program emulates IBM 3278 display terminals (models 1 to 5) and IBM 3279 display terminals (models 2A, 2B, 3A, 3B). It supports EHLLAPI programs and file transfer between your local system and IBM host systems.

This chapter includes information on:

- `sun3270x` invocation
- `sun3270x` command-line keywords
- `sun3270x` display window layout
- `sun3270x` menu selections
- `sun3270x` status line
- `sun3270x` data entry
- Example login to an IBM host application

Subsequent chapters describe customizing your sun3270x display, mapping your keyboard, transferring files to and from an IBM host, and printing IBM host files.

3.1 Starting sun3270x

sun3270x is an OpenWindows application. You can invoke this application from the command line, from a script, or as a menu option. Many sun3270x programs can run in parallel; each sun3270x controls one display window and one session with an IBM host application.

Use the `-h host_name` option to attach a sunpu2.1 SNA server running on a different system in your local area network.

3.1.1 Dependencies

sun3270x accesses the keyboard map file during startup. The default keyboard map file name is `sunkeMap`. sun3270x searches the directories listed in your `PATH` environment variable to find `sunkeMap`. If the search fails, sun3270x announces that it will use an internal default map.

sun3270x attaches to the sunpu2.1 SNA server (sunpu2.1) using the `sun_pu2_espd` service name. This entry must exist in the `/etc/services` file. Normally, the installation procedure makes this entry. If sun3270x is running on a remote system, this service name must have the same port value on both the local and remote systems.

If you want sun3270x to support EHLLAPI programs, an entry must exist in the `/etc/services` file that identifies the service name for the EHLLAPI session. Normally, the installation procedure makes this entry. If sun3270x is running on a remote system or a network name server (for example, NIS) is used, you may have to update the `/etc/services` file. EHLLAPI service names begin with `sunehllapi` and end with the EHLLAPI session name; therefore, the “A” EHLLAPI session service name is `sunehllapiA`. Since sun3270x supports EHLLAPI session names “A” through “Z,” “0” through “9,” and “a” through “z,” up to 62 EHLLAPI programs can run on one system.

♦ **To start sun3270x as an EHLLAPI server, use the `-e` keyword.**

Note – The installation procedure only adds service names “A” through “H” to the `/etc/services` file.

3.1.2 Keywords

Different keywords control the operation of `sun3270x`:

- Logical Unit (LU) attachment
- Keyboard map file path name
- EHLLAPI session name
- IBM color scheme
- Window color scheme
- Window attributes
- Generic X keywords
- Miscellaneous functions

All the `sun3270x` keywords have parallel entries in the `SUNWopc1` resources file. `sun3270x` reads the `SUNWopc1` resources file during startup; you can update this file to customize your `sun3270x` behavior (see Chapter 4, “Customizing `sun3270x`” for more information). The resource name for each keyword is under the keyword and is prefixed with a '.' (period).

3.1.3 LU Attachment Keywords

The following keywords describe the LU attachment:

- `sun3270x` requests that the LU services manager connect to the `sunpu2.1` SNA server located on the named host. If the `-h` option is omitted, the local host is called.
 - h *host_name*
 - .hostName: *host_name*

- `sun3270x` requests that the LU services manager connect to the named LU (`lu_name`). The LU must be listed in the `sunpu2.1` SNA server's configuration and the `lotype` parameter must be configured to equal 2. If the `lu_name` option is omitted, the first available LU is selected (active LUs are chosen first, inactive LUs are chosen next—only LUs with `lotype=2` can be chosen).
`-l lu_name`
`.luName: lu_name`
- `sun3270x` requests that the SNA server assign it to an LU belonging to the named Physical Unit (PU). The PU must be configured in the local configuration file. If the `pu_name` option is omitted, the PU associated with the named / selected LU is used.
`-p pu_name`
`.puName: pu_name`
- `sun3270x` requests that the LU services manager connect to the LU with the specified `port_number` on the `sunpu2.1` SNA server (only if the `lotype` parameter for the LU is configured to equal 2). `Port_number` begins at 2 and corresponds to the `LOCADDR` of the LU directive in the `sunpu2.1` SNA server configuration. This option can only be specified if the `-p` option is chosen. If the `-n` option is omitted and the `-p` option is specified, the first available LU on the specified PU is connected (active LUs are chosen first).
`-n port_number`
`.portNumber: port_number`

3.1.4 Keyboard Map Attributes

The keyboard map file can be specified as:

- The path name of the `sun3270x` keyboard map file. If the `-k` option is omitted, `sun3270x` searches the directories listed in your `PATH` environment variable for a file named `sunkeMap`.
`-k key_map_pathname`
`.kbdMapFilename: key_map_pathname`

- The path name of the sun3270x keyboard file. If the -Z option is omitted, sun3270x searches the directories listed in your PATH environment variable for a file named sunkeKbd.

-Z *keyboard_pathname*

.kbdFilename: *keyboard_pathname*

- The directory name of the sun3270x keyboard file. The default is kbds. The sun3270x examines all files in this directory for a keyboard file that matches the X server description of the display system keyboard.

-Z *keyboard_directory_name*

.kbdDirname: *keyboard_directory_name*

Override sun3270x keyboard file definition with current *keysym* values. When this option is in effect, you should specify a keyboard file name (-Z or .kbdFilename) or the default keyboard (sunkeKbd) should see your current keyboard. Use this option if you use *xmodmap()* or if you alter the Xserver environment, after mapping your keyboard.

-sniffKbd

.sniffKbd: on | off

Note – Pacific Rim users, see Appendix F, “DBCS,” for additional keywords.

3.1.5 EHLLAPI Session Name

The EHLLAPI session name is specified as:

-e *ehllapi_name*

.ehllapiName: *ehllapi_name*

sun3270x takes *ehllapi_name* as the name EHLLAPI programs use to reference sun3270 display and keyboard buffers. The *ehllapi_name* must be one character in the range “A” through “Z,” “0” through “9,” or “a” through “z.” You can also specify the EHLLAPI session name in the sunpu2.1 SNA server configuration (see the *SunLink SNA 9.1 PU 2.1 Server Configuration Guide*).

3.1.6 Color Display Attributes

You can define what colors to use when representing the IBM color scheme for 3270 displays. Use the X color names or define the color with red-green-blue (RGB) weights, following the `XParseColor()` conventions. Note that you can choose the color of the 3270 display background. The IBM color keywords are listed in Table 3-1.

Table 3-1 IBM Color Keywords and Display Attributes

IBM Color Index	The default X Color	Keyword
1	medium slate blue	<code>-ibmBlue color</code> <code>.ibmBlue: color</code>
2	red	<code>-ibmRed color</code> <code>.ibmRed: color</code>
3	magenta	<code>-ibmPink color</code> <code>.ibmPink: color</code>
4	green	<code>-ibmGreen color</code> <code>.ibmGreen: color</code>
5	cyan	<code>-ibmTurq color</code> <code>.ibmTurq: color</code>
6	yellow	<code>-ibmYellow color</code> <code>.ibmYellow: color</code>
7	white	<code>-ibmWhite color</code> <code>.ibmWhite: color</code>
8	black	<code>-ibmBlack color</code> <code>.ibmBlack: color</code>
9	dark slate blue	<code>-ibmDeepBlue color</code> <code>.ibmDeepBlue: color</code>
10	orange	<code>-ibmOrange color</code> <code>.ibmOrange: color</code>
11	violet	<code>-ibmPurple color</code> <code>.ibmPurple: color</code>
12	pale green	<code>-ibmPaleGreen color</code> <code>.ibmPaleGreen: color</code>

Table 3-1 IBM Color Keywords and Display Attributes (Continued)

IBM Color Index	The default X Color	Keyword
13	pale turquoise	-ibmPaleTurq <i>color</i> .ibmPaleTurq: <i>color</i>
14	gray	-ibmGray <i>color</i> .ibmGray: <i>color</i>
15	white	-ibmWhiteFF <i>color</i> .ibmWhiteFF: <i>color</i>
16	black	-ibmBackground <i>color</i> .ibmBackground: <i>color</i>
	Replies to the host Read Partition Query/Query List structured field to indicate that eight colors are supported. Some host applications require this response to use color support.	-colorMode8 .colorMode8: on off

3.1.7 Displaying Window Color Functions

This section lists the command line options that can be run to invoke sun3270x. These options also explain the behavior of a displayed window.

You can change color functions of the display window matte, menu options, and pop-up windows:

- Start the emulator iconified.
-iconic
.iconic: on|off
- Run sun3270x in monochrome mode. For monochrome consoles, sun3270x automatically runs in black and white. For color consoles, the defaults colors are green and black.
-m
.monoMode: on|off
- When in monochrome mode, use reverse video.

```
-r
.reverseVideoMode: on|off
```

When in monochrome mode, use the color specified for the IBM color index as the foreground color.

```
-F ibm_color_index
.monoModeForeground: ibm_color_index
```

When in monochrome mode, use the color specified for the IBM color index as the background color.

```
-B ibm_color_index
.monoModeBackground: ibm_color_index
```

Use the color specified for the IBM color index as the matte color. The default index is 1 (blue).

```
-M ibm_color_index
.matteForeground: ibm_color_index
```

The color of the main menu bar. The default is X color: light blue.

```
-menuBarBackground color
.menuBarBackground: color
```

The color of the pop-up window menu bars. The default is X color: dark green

```
-popupMenuBarBackground color
.popupMenuBarBackground: color
```

The color marking top of button. The default is X color: white

```
-xtsunBevelLightColor color
.xtsunBevelLightColor: color
```

The color marking bottom of button. The default is X color: black

```
-xtsunBevelDarkColor color
.xtsunBevelDarkColor: color
```

The color of button when selected. The default is X color: gray

```
-xtsunButtonSelectedColor color
.xtsunButtonSelectedColor: color
```

The color of text in buttons. The default is X color: black

```
-xtsunButtonTextColor color  
.xtsunButtonTextColor: color
```

The color of text read out areas. The default is X color: black

```
-xtsunReadoutTextColor color  
.xtsunReadoutTextColor: color
```

The color of toggle selection box when chosen. The default is X color: red

```
-xtsunToggleSelectedColor color  
.xtsunToggleSelectedColor: color
```

The color of pop-up window background. The default is X color: light blue

```
-xtsunPopupWindowBackground color  
.xtsunPopupWindowBackground: color
```

3.1.8 Main Display Window Attributes

This section contains the command line options that can be run to invoke sun3270x. These options also explain the attributes of a displayed window.

You can change attributes of the main display window and the pop-windows:

The default geometry is 700x500+100+100. Geometry format is defined by XParseGeometry().

```
-geometry widthxheight+Xdisplacement+Ydisplacement  
.geometry: widthxheight+Xdisplacement+Ydisplacement
```

Name in the window's title bar. The default title is the name of the sun3270x invocation.

```
-T title_name  
-title title_name  
.title: title_name
```

Display no main menu bar.

```
-noMbar  
.noMenuBarMode: on|off
```

Display no Options button on the main menu bar.

```
-noOptions
.noOptions: on|off
```

Path name of the icon file to use when the sun3270x window is closed.

```
-iconFilename filename
.iconFilename: filename
```

Type of X-cursor in the main window. The default cursor is `top_left_arrow`.

```
-cursor cursor_name
.cursor: cursor_name
```

Type of X-cursor in pop-up windows. The default cursor is `top_left_arrow`.

```
-popupCursor cursor_name
.popupCursor: cursor_name
```

The function of the Caps Lock key affects only alphabetic keys or all keys.

```
-capsLockAlphaOnly
.capsLockAlphaOnly: on|off
```

Name in the keyboard mapper title bar. The default title is the name of the sunke invocation.

```
-sunkeTitle title_name
.sunkeTitle: title_name
```

Path name of the icon file to use when the sunke window is closed.

```
-sunkeIconFilename filename
.sunkeIconFilename: filename
```

3.1.9 X Keywords

Besides the geometry and title keywords listed above, sun3270x also supports several generic X keywords that are passed to the X server:

Pass resource strings directly to the X server.

```
-xrm resource_string
```

Identify the instance of sun3270x to the X server. The *instance_name* is used to more specifically identify resource values in the SUNWopc1 resources file that you want associated with this invocation sun3270x. Use this construct to identify a group of resource values for connecting to the sunpu2.1 SNA server and identifying the EHLLAPI session name, for example:

```
-name instance_name
.name: instance_name
```

Inform the X server to display the sun3270x emulation window on a particular workstation console or X-term. *display_name* has the following format — *host_name:0.0*. Display format is defined by XOpenDisplay().

```
-display display_name
.display: display_name
```

3.1.10 Miscellaneous Functions

The following miscellaneous functions can be specified at the command line:

- Use base color support (green and white (intensified)).
-b
.baseColorMode: on|off
- Use only uppercase characters in the 3270 emulation window.
-s
.monoCaseMode: on|off
- If the requested LU or an LU Type 2 is not immediately available from the sunpu2.1 SNA server, the emulator terminates. Use the following command to specify this function.
-i
.immediateMode: on|off
- Disable the audible bell function.
-noBell
.noBell: on|off

- Disable automatic keyboard grab on mouse entry into the emulation window.
 - noGrab
 - .noGrab: on|off
- Initiate a heartbeat with the Xserver. If this value of consecutive heartbeats is missed from the Xserver, terminate the Xserver connection and terminate.
 - heartbeat
 - .heartbeat: *value*
- Request SUNWopc1 to print the ASCII/EBCDIC translation tables during initialization. See Chapter 4, “Customizing sun3270x,” for the file format.
 - P
 - .printTrans
- Identify the file that contains the appropriate ASCII/EBCDIC translation tables. See Chapter 4, “Customizing sun3270x,” for the file format.
 - a *filename*
 - .asciiEbcDicTableFilename: *filename*
- Identify the file that contains the appropriate ASCII/EBCDIC translation tables and request that sun3270x print the tables during initialization. See Chapter 4, “Customizing sun3270x,” for the file format.
 - A *filename*
 - .asciiEbcDicTableFilenameAndDump: *filename*
- Request sun3270 to print the ASCII/EBCDIC translation tables used by EHLLAPI during initialization. See Chapter 4, “Customizing sun3270x,” for the file format.
 - U
 - .printEhllapi
- Identify the file that contains the appropriate ASCII/EBCDIC translation tables to use for EHLLAPI. See Chapter 4, “Customizing sun3270x,” or the file format.
 - y *filename*
 - .asciiEbcDicEhllapiFilename: *filename*
- Identify the file that contains the appropriate ASCII/EBCDIC translation tables for EHLLAPI and request that sun3270x print the tables during initialization. See Chapter 4, “Customizing sun3270x,” for the file format.

- Y *filename*
.asciiEbcdicTableEhllapinameAndDump: *filename*
- Identify the base file name to write screen output to when the dump screen option is selected in series file name mode. Each dump screen operation creates a new file with the *base_filename* and a numeric suffix (incremented for each new operation). The default: sun3270ScreenDump.
 - d *base_filename*
.dumpScreenSeriesFilename: *base_filename*
- Identify the file name to write screen output to when the dump screen option is selected in fixed file name mode. The default: sun3270ScreenDump.
 - D *filename*
.dumpScreenFixedFilename: *filename*
- Identify the default dump screen mode. The default: Fixed.
 - dumpScreenMode Fixed|Pipe|Interactive|Series
.dumpScreenMode: Fixed|Pipe|Interactive|Series
- Identify the default dump screen pipe command.
 - dumpScreenPipe *pipe-command*
.dumpScreenPipe: *pipe-command*
- Show sun3270x usage.
 - u
.usageMode: on|off

- Trace mode of operation. The supplied value for the *trace_flag* is used to determine which internal traces are captured in the sun3270x trace log for debugging. The bits in the *trace_flag* value indicate which trace to activate. The value of the *trace_flag* may be specified according to the syntax recognized by `strtoul(3)`. The defined traces and associated *trace_flag* values are as follows:

```
-t trace_flag
.traceFlag: trace_flag
```

Trace files are created in `/tmp` and named `sunlib_pid` and `sunlib_pid.1`. Traces accumulate in `sunlib_pid` until 1000 trace points have been logged. Then, this file is renamed `sunlib_pid.1` and tracing continues in `sunlib_pid`.

```
0x0001      trace buffers to/from SNA Server API
0x0002      trace buffers from the keyboard and to the screen
0x0004      trace information on internal Finite State
              transitions.
```

- When sun3270 settings are saved in a resource file, selecting this option applies the changed settings into the current operation.


```
-saveImpliesApplyMode
.saveImpliesApplyMode
```
- Causes sun3270 to preserve the current layout of any resource files, and simply updates the changed resources. This is helpful for commented resource files.


```
-mergeResourcefile
.mergeResourcefile
```
- The left mouse button controls the cursor position.


```
-leftMouseCurPostionMode
-leftMouseCurPostionMode
```
- Enables you to cut and paste lines into the text at the current cursor position. The pasted text is inserted, moving the subsequent text over.


```
-cpasteInsertMode
.cpasteInsertMode
```
- Toggles the insert key between text insertion and text overwrite.


```
-G
.toggleInsertMode
```

3.1.11 File Transfer Menu Defaults

The functions associated with File transfer menu and defaults are described below:

- Find the host application used for file transfer. The default: TSO
-ftApplName TSO|VMS|CICS
.ftApplName: TSO|VMS|CICS
- Identify the Unix system file name for file transfer operations to/from the TSO host. See Chapter 6, “Using File Transfer,” for more information on this parameter.
-ftTsUnixFilename *Unix system file name*
.ftTsUnixFilename: *Unix system file name*
- Identify the host system file name for file transfer operations to/from the TSO host. See Chapter 6, “Using File Transfer,” for more information on this parameter.
-ftTsHostFilename *Unix system file name*
.ftTsHostFilename: *Unix system file name*
- Identify the host system file name for file transfer operations to/from the VMS host. See Chapter 6, “Using File Transfer,” for more information on this parameter.
-ftVmHostFilename *Host system file name*
.ftVmHostFilename: *Host system file name*
- Identify the host system file name for file transfer operations to/from the CICS host. See Chapter 6, “Using File Transfer,” for more information on this parameter.
-ftCicsHostFilename *Host system file name*
.ftCicsHostFilename: *Host system file name*
- Specify the password used for TSO data set password protection. See Chapter 6, “Using File Transfer,” for more information on this parameter.
-ftTsPassword *Password*
.ftTsPassword: *Password*
- Specify the direction of the file transfer operation. The default: Send
-ftTsSendReceive Send|Receive
.ftTsSendReceive: Send|Receive

- Specify the transfer property for the receiving file. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsCreateAppend Create|Append
 - .ftTsCreateAppend: Create|Append
- Specify the transfer mode for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsBinaryAscii Binary|ASCII
 - .ftTsBinaryAscii: Binary|ASCII
- Specify the CRLF for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsCrLfNocrlf Crlf|Nocrlf
 - .ftTsCrLfNocrlf: Crlf|Nocrlf
- Specifies the desired logical record length for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsLrecl *logical record length*
 - .ftTsLrecl: *logical record length*
- Specify the desired block size for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsBlksize *block size*
 - .ftTsBlksize: *block size*
- Specify the desired space allocation for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsSpace *space(prim<,sec>)*
 - .ftTsSpace: *space(prim<,sec>)*
- Specify the desired record format for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsRecfm Fixed|Variable|Undefined
 - .ftTsRecfm: Fixed|Variable|Undefined
- Specify the desired allocation unit for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftTsUnits Blocks|Tracks|Cylinders
 - .ftTsUnits: Blocks|Tracks|Cylinders
- Specify the direction of the file transfer operation. The default: Send

- ftVmSendReceive Send|Receive
 - .ftVmSendReceive: Send|Receive
- Specify the transfer property for the receiving file. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftVmCreateAppend Create|Append
 - .ftVmCreateAppend: Create|Append
- Specify the transfer mode for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftVmBinaryAscii Binary|ASCII
 - .ftVmBinaryAscii: Binary|ASCII
- Specify the CRLF for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftVmCrLfNocrlf Crlf|Nocrlf
 - .ftVmCrLfNocrlf: Crlf|Nocrlf
- Specify the desired logical record length for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftVmLrecl *logical record length*
 - .ftVmLrecl: *logical record length*
- Specify the desired record format for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftVmRecfm Fixed|Variable|Undefined
 - .ftVmRecfm: Fixed|Variable|Undefined
- Specify the direction of the file transfer operation. The default: Send
 - ftCicsSendReceive Send|Receive
 - .ftCicsSendReceive: Send|Receive
- Specify the transfer property for the receiving file. See Chapter 6, “Using File Transfer,” for more information on this parameter.
 - ftCicsCreateAppend Create|Append
 - .ftCicsCreateAppend: Create|Append

- Specify the transfer mode for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
`-ftCicsBinaryAscii Binary|ASCII`
`.ftCicsBinaryAscii: Binary|ASCII`
- Specify the CRLF for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
`-ftCicsCrLfNocrlf Crlf|Nocrlf`
`.ftCicsCrLfNocrlf: Crlf|Nocrlf`
- Specifies the desired comment string for the file transfer operation. See Chapter 6, “Using File Transfer,” for more information on this parameter.
`-ftCicsComment comment string`
`.ftCicsComment: comment string`

3.1.12 Light Pen Defaults

You can use the following defaults for the Light Pen - Cut/Copy - Paste pop-window:

- Specify if the Light Pen - Cut/Copy - Paste pop-window will be displayed when the right mouse button is pressed.
`-noCPP`
`.noCPP: on|off`
- Specify the default mode for all mouse operations. See Section 3.11, “Special Functions,” for more information on this parameter.
`-cpPenFieldAll Pen|Field|All`
`.cpPenFieldAll: Pen|Field|All`
- Specify the default mode for cut operations. See Section 3.11, “Special Functions,” for more information on this parameter.
`-cpLineRect Line|Rectangle`
`.cpLineRect: Line|Rectangle`
- Specify using the mouse for copy and paste functions. The left mouse button is used to select the copy text. The middle mouse button is used to paste.
`-cpasteMouse`
`.cpasteMouse: on|off`

3.1.13 Host Graphics Keywords

You can change the defaults for host graphics:

- Chooses whether or not to enable GDDM support during this run of the sun3270. If GDDM support is enabled, Query replies to the host indicate that GDDM and the GDDM datastream will be fully supported. If GDDM support is not enabled, Query replies to the host indicate NO GDDM support and any GDDM datastream data will be silently ignored. The default is OFF.

```
-gddm
.gddmMode      on | off
```

- This flag is similar to `-gddm/.gddmMode` described above, except that it causes sun3270 to allocate a separate set of 3 planes in pixel memory for the Graphic Layer. On some hardware this yields faster screen updates and refreshes in both the Text Layer and the Graphics Layer. It yields smoother, real-time updates of the Graphics Layer as GDDM data stream arrives. This mode of GDDM is required for GDDM mix modes OR and XOR, for supporting 3-plane LPSs in Graphics Layer, and for the GDDM crosshair graphics cursor. If this mode is not enabled, mix modes OR and XO will act the same as mix mode OVERPAINT, Query responses to the host will indicate no support for 3-plane LPSs, any Load PS structured field having a color field of other than 0 will be rejected, and the GDDM crosshair graphics cursor will be unavailable (only the cross graphics cursor will be available). If both the `-gddm` and `-gddmDirect` are given, `-gddmDirect` takes precedence. The default is OFF.

```
-gddmDirect
.gddmDirectMode      on | off
```

- The type of X-cursor to use for the cross-style GDDM graphics cursor. The default is crosshair.

```
-graphicCursor  cursor_name
.graphicCursor  cursor_name
```

- Chooses whether or not to enable support for the Load Programmed Symbols structured field during this run of the sun3270. If enabled, Query replies to the host indicates support for Load PS and such structured fields will be processed when seen. If not enabled, Query replies to the host

indicates NO support and any Load PS structured fields seen will be ignored. This enables/disables Load PS for both the Text Layer and the Graphics Layer. The default is OFF.

```
-loadps
.loadpsMode      on | off
```

- Path name of the bitmap file to read and use for GDDM standard fill pattern number *i*. Values for *i* are 1 - 14. The default is `gddmPati`. Examples: `gddmPat1`, `gddmPat5`, `gddmPat12`, etc.

```
-gddmPati      filename
.gddmPati      filename
```

- Causes the sun3270 to consider GDDM datastream coordinate value (0, 0) to be the upper-leftmost pixel in the Graphics Layer and so forth rather than considering (0, 0) to be the center of the Graphics Layer, which is the normal interpretation. A small number of hosts do consider datastream (0, 0) to be the upper-left pixel in the Graphics Layer, hence this switch. The default is OFF.

```
-gddmUpperLeft
.gddmUpperLeftMode      on | off
```

- Enables support of the WCC(Reset) bit (WCC bit 1). If enabled, the sun3270 performs reset actions described in the *IBM 3270 Information Display Station Data Stream Programmer's Reference* when the WCC(Reset) bit is on in the WCC of an `Erase_Write` or `Erase_Write_Alternate` command. (This bit is ignored on `Write` and `Erase_all_Unprotected` commands.) If not enabled, the WCC(Reset) bit is always ignored. The default is OFF.

```
-R
.wccReset      on | off
```

Note – This bit will be enabled if gddm support is enabled. When running sun3270 with GDDM support, it functions as an 8-color device.

3.1.14 User Button and Function Keywords

You can use the following defaults for the User buttons and User functions within the Options menu.

- Specify the default title string for user button 1. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton1Title *User Button Title string*
.userButton1Title: *User Button Title string*
- Specifies the default command string for user button 1. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton1Command *User Button Command string*
.userButton1Command: *User Button Command string*
- Specify the default title string for user button 2. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton2Title *User Button Title string*
.userButton2Title: *User Button Title string*
- Specify the default command string for user button 2. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton2Command *User Button Command string*
.userButton2Command: *User Button Command string*
- Specify the default title string for user button 3. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton3Title *User Button Title string*
.userButton3Title: *User Button Title string*
- Specify the default command string for user button 3. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton3Command *User Button Command string*
.userButton3Command: *User Button Command string*
- Specify the default title string for user button 4. See Section 3.11, “Special Functions” for more information on this parameter.
-userButton4Title *User Button Title string*
.userButton4Title: *User Button Title string*
- Specify the default command string for user button 4. See Section 3.11, “Special Functions” for more information on this parameter.

-userButton4Command *User Button Command string*
 .userButton4Command: *User Button Command string*

- Specify the default title string for user button 5. See Section 3.11, “Special Functions” for more information on this parameter.

-userButton5Title *User Button Title string*
 .userButton5Title: *User Button Title string*

- Specify the default command string for user button 5. See Section 3.11, “Special Functions” for more information on this parameter.

-userButton5Command *User Button Command string*
 .userButton5Command: *User Button Command string*

You can set defaults for the user exits on LU activation and deactivation:

- Specify the default command string for LU activation. The format of the command is identical to the user button command string.

-actCommand *LU activation command string*
 .actCommand: *LU activation command string*

- Specify the default command string for LU deactivation. The format of the command is identical to the user button command string.

-dactCommand *LU deactivation command string*
 .dactCommand: *LU deactivation command string*

3.1.15 sun3270x Command Line Configuration Examples

- To attach a sunpu2.1 SNA server on a remote system named bruno, type:

```
% sun3270x -h bruno
```

- To attach to an LU named BLU01101 (on the local system) and serve as the EHLLAPI presentation space named A, type:

```
% sun3270x -l BLU01101 -e A
```

- To change IBM color blue to the X-color light blue and display the sun3270x window on the remote system named bruno, type:

```
% sun3270x -ibmBlue "light blue" -display "bruno:0.0"
```

- If you have set up your SUNWopcl resources file and you want to identify resource values associated with instance name D1, type:

```
% sun3270x -name D1
```

3.2 sun3270x Window Layout

The sun3270x window can be displayed on all systems that support an X server. It inter-operates with OpenWindows, and X Window Manager window managers. The sun3270x window can be moved and resized; it can be closed (iconified) and then re-opened.

On color consoles, sun3270x uses all sixteen IBM colors; on monochrome consoles, sun3270x runs in black and white. You can choose the colors and operational characteristics of the sun3270x window by using the SUNWopcl resources file to describe the appropriate resource values. The SUNWopcl resources file is detailed in Chapter 4, "Customizing sun3270x."

Figure 3-1 illustrates the basic components of the sun3270x window.

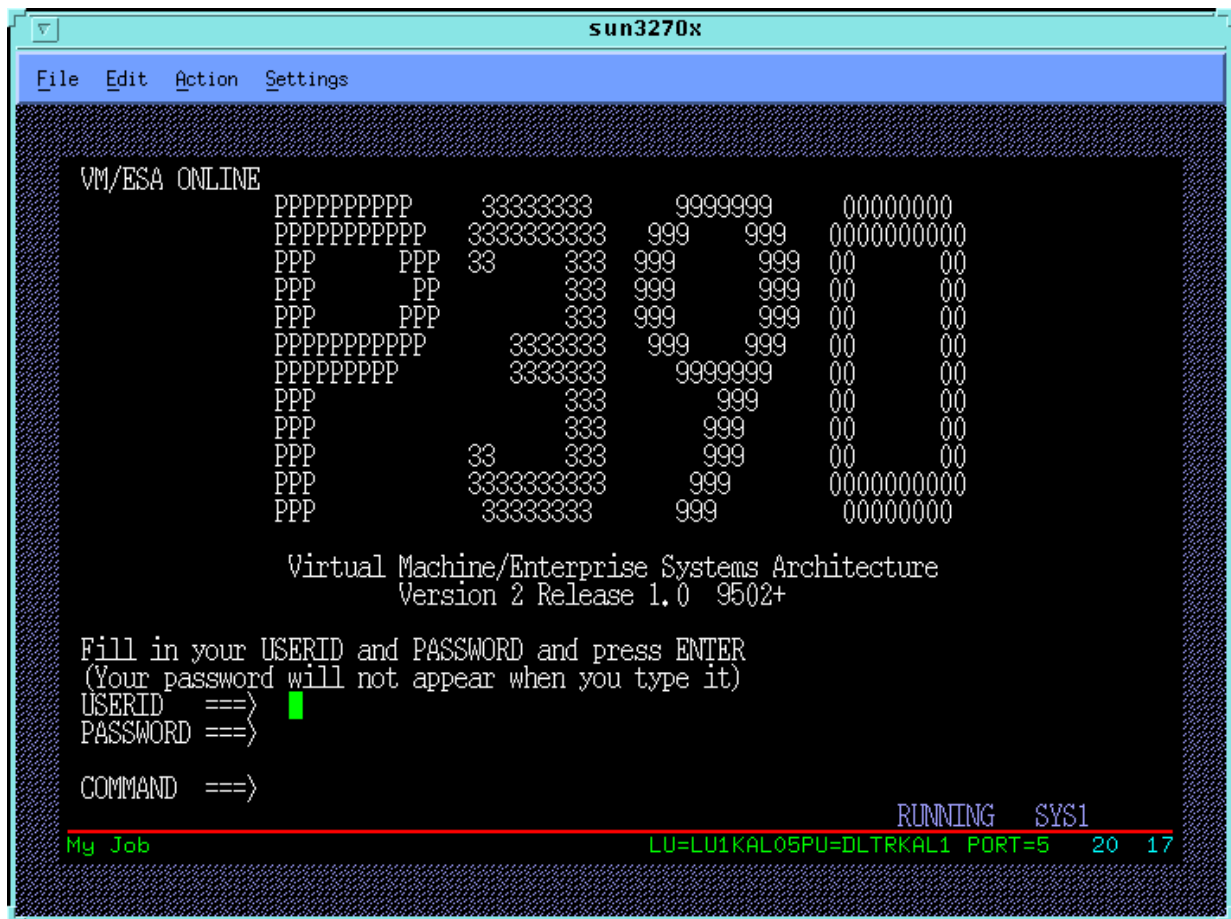


Figure 3-1 The sun3270x Window

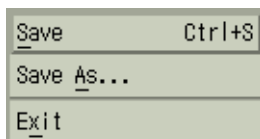
The sun3270x window is composed of four major parts:

- **Window Frame** — Contains the generic title bar and window border provided by the system window manager.
- **Main Menu** — The main menu enables you to choose sun3270x options like displaying the keyboard mapping, executing file transfers, and changing sun3270x settings dynamically. A quit button also enables for simple exit.

- **Matte** — Matte is the textured background surrounding the 3270 emulation screen display. A matte varies in size depending on the dimensions of the window and the available font sizes. sun3270x operates in a “best fit” mode — as you expand or shrink the window, sun3270x fills the interior of the window with the 3270 emulation screen display.
- **3270 Emulation Screen** — The 3270 emulation screen displays data transmitted by IBM host applications and accepts your keyboard input. The 3270 emulation screen also contains an Operator Information Area (OIA) or Status Line at the bottom of the screen.

3.2.1 File Menu

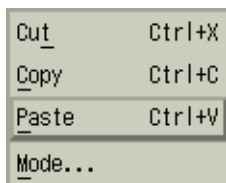
Pressing the File button displays the following menu. Each function is described below.



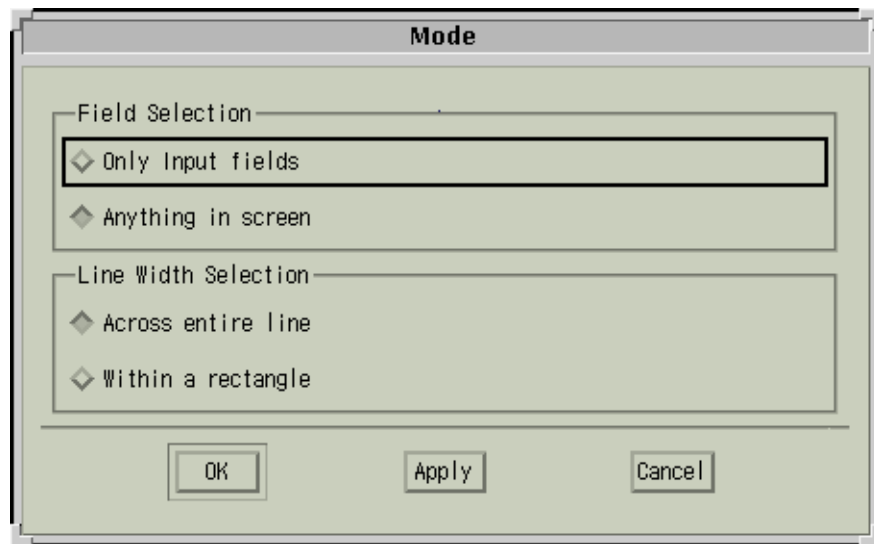
- **Save** — Save changes to an open session at any time without closing the session. If the session does not have a name, the screen displays the SaveAs dialog box.
- **SaveAs** — Saves the current session to a new or different name. The screen displays the SaveAs dialog box. Enter the new name and press OK.
- **Exit** — Exits the sun3270x session.

3.2.2 Edit Menu

Pressing the Edit button displays the following menu which enables you to cut, copy, and paste text as well as set up Edit Mode.



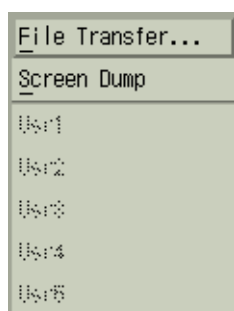
- **Mode** — Displays the Mode dialog box. The options are described below.



- **Only Input Fields** — You can only cut or copy characters in data entry fields. Each field is transferred as a single line. Fields spanning multiple lines are transferred as multiple lines. When this field is selected, all other options are excluded. Use this mode when transferring fields from one sun3270 display to another, or to capture only the input field data.
- **Anything in Screen** — You can only cut or copy all characters in the selected area. Field characters are converted into space characters. Each screen line is terminated by a new line.
- **Across Entire Line** — Select characters across entire lines. This mode parallels the normal Cut/Copy functions in Windows.
- **Within a Rectangle** — Use a rectangle as the selection constraint. This mode enables you to transfer “rectangles” of 3270 display data.

3.2.3 Action Menu

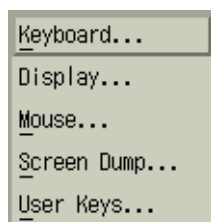
Pressing the Action button displays the following menu. Each function is described below.



- **File Transfer** — Transfers files between your local system and the IBM host. Chapter 6, “Using File Transfer,” discusses the various file transfer options.
- **Screen Dump** — Activates a Screen Dump request. See “Settings: Dynamic SUNWopcl Resource Updates” on page 4 for how to change the Screen Dump request.
- **Usr1, Usr2, etc.** — User Keys. When set, these menu items do the same pressing a user-defined function key. See “Settings: Dynamic SUNWopcl Resource Updates” on page 4 for how to change user-defined keys.

3.2.4 Settings Menu

Pressing the Settings button displays the following menu that enables you to dynamically change certain attributes of the sun3270x display and dump screen options. Chapter 4, “Customizing sun3270x,” describes all the features that you can change.



3.3 Stopping sun3270x

You can terminate the sun3270x display in several ways:

Title Bar Menu — Most window managers provide a Quit function in the menu associated with the window frame.

Stop the sun3270x process.

File Menu — Press File->Exit to terminate the session.

3.4 Status Line

The current status of the emulated 3270 display station is continuously displayed on the last line of the sun3270x 3270 emulation screen (below the horizontal bar).

The Status Line in Figure 3-2 shows that the user is currently in an SSCP-LU session (System Operator); the host is busy processing (Wait); the display is in Insert mode; the LU is SLU02 (port 2 on PU SP63) and the EHLLAPI session name is A; and the cursor is on row 4, column 56.

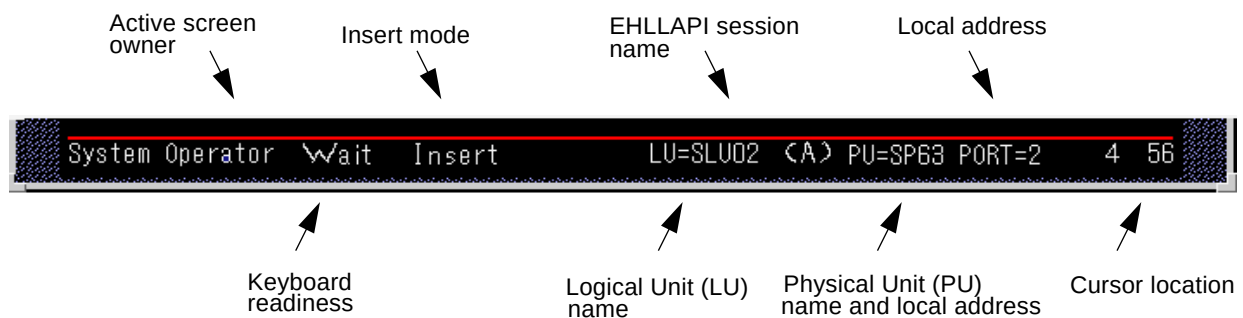


Figure 3-2 Sample Status Line

The components of the Status Line are defined in Table 3-3.

Table 3-2 Status Line Meaning

Indicator	State
Status	The current screen owner/active session is displayed.
	Inactive LU The SSCP-LU session was not established by the host.
	System Operator The SSCP-LU session owns the display.
	My Job The LU-LU session owns the display and the terminal emulator is bound in session to a primary LU.
	Unowned The LU-LU session owns the display, but the terminal is not bound in session to a primary LU.
	Disconnected SUNWopc1 cannot connect to the sunpu2.1 SNA server.
Readiness	The keyboard input capability is displayed.
	Blank Keyboard input is enabled.
	Wait Keyboard input is disabled.
Modes	The terminal insert mode is displayed.
	Blank The terminal emulator is not in insert mode.
	Insert The terminal emulator is in insert mode.
LU	The current Logical Unit name is displayed. If sun3270x supports an EHLLAPI session, the EHLLAPI session name is displayed in parenthesis after the Logical Unit name.
PU	The current Physical Unit name is displayed.
PORT	The current PU port number is displayed.
Cursor	The current location of the cursor (row/column).

3.5 *Data Entry Overview*

All `SUNWopcl` programs (`sun3270x`, `sun3270tty`, `suntn3270x`, and `suntn3270tty`) share common data entry mechanisms. Except where noted, the following sections apply to each of these programs.

Since your keyboard probably differs from an IBM 3270 display station's keyboard, `SUNWopcl` provides a configurable mapping capability to associate your keystrokes with IBM keystrokes. For `sun3270x` and `suntn3270x`, see Chapter 5, “Using Keyboard Mapper,” for directions on mapping your keyboard to provide comparable 3270 keyboard input functions. For `sun3270tty`, and `suntn3270tty` see Appendix B, “Mapping `sun3270tty` Keyboards.”

3.5.1 *SunLink 3270 Screen Formats*

A SunLink 3270 client display has two types of screens: unformatted and formatted. A formatted screen has fields defined by an IBM host application. The basic field types are:

- **Protected**—A protected field contains information from the host, which you cannot overwrite. It is “protected” from user input. If you enter data in a protected field, the keyboard will lock and you have to press the IBM reset key to restore the keyboard for input.
- **Unprotected**—An unprotected field enables you to insert information.
- **Non-display**—A non-display field does not echo input as it is entered (for example, password fields).
- **Autoskip**—An autoskip field is “skipped” over by the cursor.

3.5.2 *3270 Keyboard Input Summary*

As you enter each character, `SUNWopcl` validates it against field attributes to ensure that the character belongs where you are entering it. If the data character is valid, it is displayed on the screen; if the data character is invalid, an alarm sounds and the cursor does not move. An example of an invalid character is a character entered in a protected field. To recover, press the IBM Reset key and enter a valid character.

The cursor indicates where the next entered character will appear on the screen. The cursor advances automatically (unless you enter an invalid character) as you enter data. If a field is not completed, use the Return, Tab, or cursor key to move the cursor to the next data entry field. Also, you can correct mistakes by positioning the cursor at the text to be changed and typing over the error. Cursor movement keys are described later in this chapter.

After entering data, you can instruct `SUNWopcl` to send it to the host program by pressing the Enter key, a function key, or an Attention key. The correct key to use depends on the requirements of the IBM host application. See the IBM host application documentation for application input requirements.

While `SUNWopcl` is transmitting your data to the host program, the keyboard is “locked.” The keyboard will not accept keystrokes until:

- Data transmission is complete
- Host system sends a message to `SUNWopcl` to be sent to your screen
- You enter the Reset function
- You enter the Log Off function
- You enter the System Request function

3.5.3 *Keyboard Functions*

Keyboard functions are divided into six categories:

1. Data entry functions — for entering data
2. Start/stop data entry — for terminating or enabling data entry, or transmitting data
3. Cursor control functions — for positioning the cursor on the screen
4. Field control functions — for formatting or structuring fields
5. Editing mode functions — for inserting and deleting characters
6. Special functions

With the exception of the Data Entry functions, these categories represent key functions that control the position of input and transmission of data to the IBM host application. Table 3-3 lists these **SUNWopc1** keys and includes a brief description of their actions. Subsequent sections detail the actions of the Data Entry keys as well as the keys listed in Table 3-3.

Table 3-3 Keyboard Functions

Function	Mnemonic	Description
Cursor Control:		
Backtab	backtab	Move cursor to previous field.
Cursor Down	down	Move cursor down one line.
Cursor Left	left	Move cursor left one character position.
Cursor Right	right	Move cursor right one character position.
Cursor Up	up	Move cursor up one line.
Home	home	Move cursor to first field.
New Line	newline	Move cursor to first field on next line.
Tab Forward	tab	Move cursor to next field.
Start/Stop Data Entry:		
Attention	attn	Request turn to send data.
Clear	clear	Clear screen.
Enter	enter	Initiate transfer of data to host.
Function keys (PF1-24)	pf1-pf24	Initiate transfer of data to host.
Program access (PA1-3)	pa1-pa3	Initiate transfer of data to host.
Reset	reset	Clear input inhibited status or insert mode.
Editing Mode:		
Delete	delete	Delete character at current cursor position.
Insert mode	insert	Insert characters at current cursor position.
Field Control:		
† Clear field	clearfield	Clear field and move cursor to beginning of field.

Table 3-3 Keyboard Functions (Continued)

Function	Mnemonic	Description
† Delete left	deletelleft	Delete character to the left of the cursor.
DUP	dup	Duplicate operation for field.
Erase field	eeof	Erase to end of the current field.
Erase input	ein	Erase all unprotected fields.
Field mark	fm	Mark the end of a field or a subfield.
Special Function:		
Systems request	sysreq	Switch to other session (i.e. LU-LU to SSCP-LU).
Cursor select	cursorl	Select the current field, if selectable.
Print	dumpscreen	Copy screen to file or program.
• Cut	cut	Cut selected portion of the screen.
• Copy	copy	Copy selected portion of the screen.
• Paste	paste	Paste selection to the current cursor position.
• User-N	user defined	Invoke user-defined command pipeline N.

† Sun special function keys. • Keys valid for sun3270x only.

3.6 Data Entry

The data entry keys are the same on all 3270 terminals. The data entry functions include:

- All alphabetic characters (upper and lowercase A through Z)
- Numeric characters (characters 0-9 located above the alphabetic keys on the main keypad, or on the numeric keypad)
- Non-alphabetic characters (such as the asterisk (*))—punctuation and special symbols located above the alphabetic keys, at the bottom and to the right of the alphabetic keys, and the numeric keypad.

- The spacebar (do not use the spacebar to position the cursor unless text is to be replaced by spaces)

Shift

Shift capitalizes all the alphabetic characters. It also causes the top legend of any other key to be entered when pressed in combination with that key.

Caps Lock

The Caps Lock key capitalizes all alphanumeric keys until the Caps Lock key is pressed again. The Caps Lock key does not affect any other keys.

3.7 Cursor Control

The Cursor Control keys position the cursor without changing the contents of the screen. These functions include:

- Moving the cursor one character position at a time
- Positioning the cursor to the first character position in a data entry field

Cursor Control is subject to limitations imposed by data field attributes defined by an application program. For example, you cannot use all the cursor-control functions to move the cursor into a protected field.

3.7.1 Moving the Cursor from Character to Character

The following cursor control keys position the cursor one character position at a time.

Cursor Down

Cursor Down moves the cursor down one line at a time in either non-data or data entry fields. If the cursor is on the last line in any column, the cursor moves to line 1 in the same column.

Cursor Left

Cursor Left moves the cursor one character position to the left in data entry fields.

- If the cursor is in the first character position of a line, the cursor moves to the last character position of the previous line.
- If the cursor is in line 1, column 1, the cursor moves to the last character position of line 24.

Cursor Right

Cursor Right moves the cursor one character position to the right.

- If the cursor is in the last character position of a line, the cursor moves from the end of the line to the first character position of the next line.
- If the cursor is in the last character position of line 24, the cursor moves to the first character position of the screen.

Cursor Up

Cursor Up moves the cursor up one line at a time in either non-data entry or data entry fields.

- If the cursor is on line 1 in any column, the cursor moves to line 24 in the same column.

3.7.2 Moving the Cursor from Field to Field

These following cursor control keys position the cursor to the first character position in a data field.

Backtab

Backtab moves the cursor to the start of the previous (non-protected, non-skipped) field.

- If a display consists of mixed (non-data entry and data entry) fields and the cursor is located either in the attribute character position of one of the mixed fields, or in the first character position of a data entry field, or in any character position of a non-data entry field, the cursor moves back to the first alphanumeric character position of the preceding data entry field.
- If the cursor is located in an alphanumeric character position (other than the first position) of a data entry field, the cursor moves back to the first alphanumeric character position in the same data entry field.

- If a display is either unformatted or consists only of non-data entry fields, the cursor moves back to the first character position on line 1 of the screen.

Home

The Home key moves the cursor to one of the following positions:

- If a display is formatted, the Home key moves the cursor to the first character position of the first data entry field on the screen.
- If a display is unformatted, the Home key moves the cursor to line 1, column 1.

New Line

New Line moves the cursor to the first character position of the next display line into which you can enter data.

- If a display consists of either mixed (non-data entry and data entry) fields or data-only entry fields, the cursor moves to the first position of the next data entry field.
- If a display consists entirely of only non-data entry fields, the cursor moves to the first character position on line 1 of the screen.
- If a display is formatted, the cursor wraps to the first character position of the next line.

Tab Forward

Tab moves the cursor forward to the first character position of the next data entry field.

If a display is either unformatted or consists only of non-data entry fields, the cursor moves forward to the first character position on line 1 of the screen.

3.8 Start/Stop Data Entry

The following function keys enable or terminate data entry from the terminal. Some functions also cause **SUNWopc1** to transmit data to the host computer.

Attention

The Attention function key interrupts the IBM host application. This function is invalid if you are using the SSCP-LU session.

Clear

- If you use the Clear function key while using the SSCP-LU session, your screen is cleared immediately.
- If you use the Clear key while in LU-LU session, your screen does not clear immediately; the keyboard is locked and only the Reset key will unlock it.

Anything but the Reset key sounds the alarm. When the host computer sends an appropriate code, the WAIT state is extinguished, and your terminal screen clears.

Enter

Enter sends the currently displayed data to the host computer and locks the keyboard. Until the keyboard is unlocked, the alarm will sound if you press a data entry key.

Function Keys (PF1 through PF24)

Function keys terminate data entry and cause SUNWopc1 to transmit the displayed data to the host computer. There are twenty four IBM 3270 function keys that are designated PF1 through PF24.

An application program defines what happens when you press a function key. Each application program can assign a separate function to each function key. For example, an application program might indicate replacement of one display data entry form with another display data entry form whenever you press the PF5 function key. Or, if you press the PF2 function key, the application program could be notified that all necessary data is entered.

If you press any of the function keys, the keyboard is locked. Until the keyboard is unlocked, the alarm sounds if you press a data entry key.

Program Access Keys

Program access keys terminate data entry and cause SUNWopc1 to transmit a message to the IBM host application.

(PA1 Through PA3)

An application program defines what happens when you press a program access key. Each application program can assign separate actions to each program access key.

If you press any program access keys, the keyboard is locked. Until the keyboard is unlocked, the alarm will sound if you press a data entry key.

Reset

Reset unlocks your keyboard so you can enter more data.

3.9 *Editing*

The following editing function keys insert and delete characters in data entry fields:

Insert Mode

Use Insert mode to enter consecutive characters into a data entry field without altering characters already there. Press the Reset key to terminate the insert mode.

When you insert text, all characters within a field from the cursor to the first NULL character are shifted right. If the cursor is located at a null (non-character) position in a data entry field and you insert a character, the character is inserted in the null position and no character shifting occurs.

The insert function does not span lines. A separate insert action must be performed on each line.

If you try to insert characters in a filled field, an alarm sounds.

In an unformatted screen, the Insert mode treats the entire screen as one field and, additionally, insert will span lines and the end-of-screen.

Delete

Delete helps delete the character at the cursor in a data entry field. The cursor does not move from its current location, and any characters to the right shifts one position to the left. Vacated character positions at the end of the fields are filled with null characters. A separate delete action must be performed for each character that is to be deleted.

3.10 Field Control

The following field control keys affect the format or structure of fields:

Clear Field

Clear Field is a Sun special function key that erases the current field and positions the cursor to the field's first character location.

Delete Left

Delete Left is a Sun special function key that positions the cursor one character to the left and deletes the character in that position.

DUP

DUP duplicates the information from the previous record in a data entry field. When you use this function, `SUNWopc1` displays an asterisk (*) and a tab function is performed.

`SUNWopc1` transmits a code (represented by the displayed asterisk) to the host application program. Upon receipt of the code, the host application program initiates a duplication operation where the asterisk (code) is located.

Erase Input

Erase Input erases all data entry fields. The cursor is repositioned to the first location of the first possible data entry field.

- If the display consists entirely of non-data entry fields, the cursor moves to line 1, column 1.
- If a display is unformatted, all character locations are cleared to null characters. The cursor moves to line 1, column 1.

Erase to End-of-Field

Erase-to-End-of-Field erases the current data entry field, from the cursor position to the end of the field. The cursor does not move from its current location.

If a display consists of non-data entry fields or if the cursor is located in an attributed character position, an alarm sounds and the keyboard locks.

Field Mark

Field Mark displays a semicolon as a field delimiter.

- If a display consists of data entry fields, subfields within a data entry field are delimited with semicolons. If a display is unformatted, the semicolon indicates the end of a field.
- If the cursor is located either in an attribute character position or in a non-data entry field, an alarm sounds and the cursor does not move.

3.11 Special Functions

The following special functions keys provide extended capabilities.

System Request

The System Request function key transfers ownership of the screen display. Display ownership is indicated by the status line, and only one session can be the display device owner at a time. During the session or the ownership period, only the device owner can display data. Any attempt by a non-owner to use the display is rejected.

If the display device is not owned, as indicated on the status line, you can use the System Request function to control which session owns the device.

You can also use the System Request function key to transfer display ownership between sessions. This transfer interrupts communications that take place for the session without waiting for completion of outbound transmissions (transmissions from the host). The inbound transmissions (transmissions to the host) normally are completed before this type of transfer takes place.

Print

Print enables you to capture information on a sun3270x screen in a file. Press the Print key to start the copy. You can change the actions of the Print key by updating the appropriate Settings under the main menu bar's Options button.

Selector Light Pen

Selector Light Pen operation enables the mouse to function as a light pen.

Cursor Select

Cursor Select enables the Selector Light Pen function to be performed from the keyboard. The Cursor Select Key may be used on any field defined as a Selector Light Pen detectable field.

Cut, Copy, and Paste

SUNWopc1 supports the Cut, Copy, and Paste functions. Due to the formatted nature of 3270 display stations, SUNWopc1 supports multiple modes for these functions. To change the Cut, Copy, and Paste mode, press the right mouse button while in the 3270 emulation screen. Figure 3-3 shows the menu that controls the selection mode.

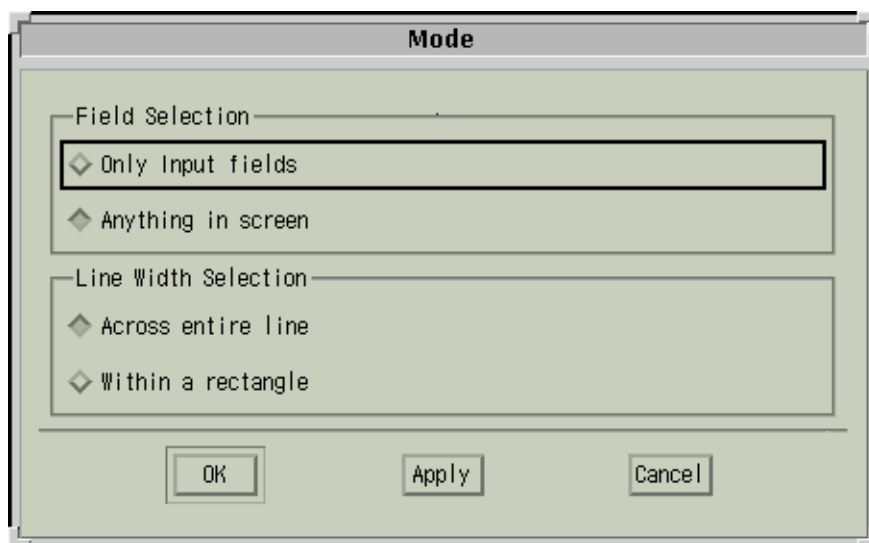


Figure 3-3 Light Pen - Cut/Copy - Paste Menu

The options are:

- **Light Pen Selection:** The left mouse button functions as a Selector Light Pen. If the left mouse button is clicked within a Light Pen selectable field, that field is selected. When this option is selected, all other options are excluded. Two types of light pen selectable fields are defined: selector fields and attention fields.
 - The selection type field is defined by a question mark (?) designator character. When selected, the selector field designation character is automatically changed to a greater than sign (>) and the Modified Data Tag bit is set (that is, this field is transmitted to the host on operator ENTER action or receipt of a Read-Modified command).
 - An attention type field is defined by a space or NULL designator character. When selected, an I/O pending attention is generated to the host. The host may then issue a Read-Modified command to retrieve the selected field.
 - A second type of attention type field is defined by an ampersand (&) designator character. When selected, an operator ENTER action is simulated on the display.
- **Only Input Fields:**
You can only Cut or Copy characters in data entry fields. Each field is transferred as a single line. Fields spanning multiple lines are transferred as multiple lines. When this field is selected, all other options are excluded. Use this mode when transferring fields from one SUNWopc1 display to another or to capture just the input field data.
- **Anything In Screen:**
Cut/Copy all characters in the selected area. Field characters are converted into space characters. Each screen line is terminated by a new line.
- **Across Entire Line:**
Select characters across entire lines. This mode parallels the normal Cut/Copy functions in X Windows.
- **Within a Rectangle:**
Use a rectangle as the selection constraint. This mode enables you to transfer “squares” of 3270 display data.

To select data, position the X-cursor and press the left mouse button. While holding the left mouse down, drag the X-cursor to the final character position you want to select. You can also select a single word by positioning the cursor within the word and double-clicking the left mouse button. A *word* is defined as a set of contiguous letters, underscores, or digits.

You can readjust the area you selected by positioning the X-cursor on the desired screen position and pressing the shift key and left mouse button simultaneously.

To de-select a selected area, press the left mouse button anywhere on the sun3270 screen.

Cut

Cut moves data from the selected area to a “logical” clipboard. The data is deleted from the sun3270x display.

Copy

Copy copies data from the selected area to a “logical” clipboard.

Paste

Paste moves data to the SUNWopcl display from the “logical clipboard” of the same or some other sun3270x display, or, from the selection highlighted in some non-sun3270x window like xterm or cmd-tool.

If the incoming data is a non-rectangular cut/copy from a SUNWopcl display, or, is from a non-SUNWopcl window, each line in the data fills in one data entry field in the SUNWopcl display starting at the position of the 3270 cursor. If the screen is unformatted, each line in the data fills one screen line.

If the incoming data is a rectangular cut/copy from a SUNWopcl display, the upper left corner of the rectangle of data is placed at the 3270 cursor and the rectangle is dropped “as is” into the SUNWopcl display, subject only to the restriction that field characters and characters within protected fields are not altered. Rectangular cut/copy-paste is useful for moving or copying data in mainframe text editors such as the TSO editor.

User-Defined Function Keys

User-defined function keys enable you to specify a command pipeline to be executed when the key is pressed. The user functions may invoke EHLLAPI programs to perform user specific tasks. The SUN3270 will not wait for the specified command pipeline to complete before continuing. Any error status returned by the specified command pipe line is reported through the Child Process Abnormal Termination status pop-up window. The child process exit status is displayed within the window.

Customizing sun3270x



The sun3270x display can be customized in a number of ways. This chapter discusses the sun3270x resources, changing sun3270x text information, and modifying the ASCII/EBCDIC translations.

If you want to customize your sun3270x keyboard mapping display, see Appendix A, “Customizing the sun3270x Keyboard.”

4.1 sun3270x Resources

sun3270x has many attributes that you can change. In X window environments, configurable attributes are called resources. The sun3270x program has many attributes that you can change. The sun3270x resources belong to the SUNWopcl class.

When the sun3270x program starts, it searches for SUNWopcl resources in order of highest precedence:

- `./SUNWopcl`
- `$XENVIRONMENT` if it doesn't exist, then `$HOME/.Xdefaults-hostname`
- Resource Manager property (xrdb), if it does not exist, then `$HOME/.Xdefaults`
- `$HOME/SUNWopcl`
- `$XAPPLRESDIR/SUNWopcl`
- `/usr/lib/X11/app-defaults/SUNWopcl`

If a `SUNWopcl` resource is found, `sun3270x` initializes the defined resource with its specified value.

In the `SUNWopcl` resources file, a '*' (asterisk) preceding a resource indicates that all `sun3270x` programs should use this resource definition, for example, `*hostname: bruno`.

Alternatively, an instance name can be used to set a resource value only when the argument `-name` to the `sun3270x` program and the instance name match. Using an instance name makes it easy to specify different `sun3270x` program activities.

A resource specified with an instance name overrides any bindings using the '*' (asterisk) paradigm. `sun3270x` command line arguments override any settings established by `SUNWopcl` resource bindings.

The following `SUNWopcl` resources exist:

- Logical Unit (LU) attachment
- Keyboard map file path name
- EHLLAPI session name
- IBM color scheme
- Window color scheme
- Window attributes
- Font pools
- Generic X keywords
- Miscellaneous functions

Most of the `SUNWopcl` resources are described in Section 3.1, "Starting `sun3270x`." All `sun3270x` resources have parallel keyword definitions except for the font pools discussed below.

♦ **To list the fonts that `sun3270x` uses to display textual data in the 3270 emulation screen, type:**

```
.fontPoolNormal: font_name1 font_name2 ... font_nameN
```

Note – `sun3270x` chooses the font used for display on a "best fit" basis. It selects the font whose width and height cover the most area within the `sun3270x` display window. The fonts must be fixed-width (character cells). If your application uses non-North American English characters, make sure that the fonts you choose contain all the ISO 8859.1-defined characters. Similarly, you need to verify the ASCII/EBCDIC translations map appropriately.

You can list the fonts that sun3270x uses to display textual information in the Status Line. sun3270x chooses the display font on a “best fit” basis. The semicolon (;) character in the resources file indicates that the rest of the line is a comment:

```
.fontPool0IA: font_name1 font_name2 ... font_nameN
```

4.2 Example SUNWopcl Resource File

Code Example 4-1 Example SUNWopcl Resource File

```
=====
;
;
; SUNWopcl
;
; %W% %G%
;
=====

;
;
; This file contains example resource entries for the sun3270x emulation
; program. When the sun3270x program starts, it searches for a SUNWopcl
; resource file in the X resource directories (e.g. $XAPPLRESDIR).
; If the SUNWopcl resources file is found, sun3270x initializes the defined
; resources with their specified values.
;
; A '*' preceding a resource, indicates all sun3270x programs should use
; this resource definition (e.g. *hostname: bruno).
;
; Alternatively, an instance name can be used to set a resource to a
; value depending on the -name argument to the sun3270x program. As
; illustrated below, using an instance name makes it easy to specify
; different sun3270x program behavior. In this example, when sun3270x
; is invoked with "-name D2", sun3270x connects to the BLU01102 Logical
; Unit and becomes the "B" EHLLAPI session.
;
; A resource specified with an instance name overrides any bindings
; using the '*' paradigm.
```

Code Example 4-1 Example SUNWopcl Resource File (Continued)

```

=====
;
;
; Note: many of the resources specified in the resource file can be overridden
;       by command line options.
;
;
*hostName:                bruno

D1.luName:                BLU01101
D1.ehllapiName:           A

D2.luName:                BLU01102
D2.ehllapiName:           B

D3.luName:                BLU01103
D3.ehllapiName:           C

*kbdMapFilename:          /usr/local/Sun/SUNWopcl/sunkeMap

*geometry:                700x500

*ibmBlue:                 hot pink

*ibmRed:                  orange

*ibmGreen:                sky blue

*ibmWhite:                turquoise

```

4.3 Settings: Dynamic SUNWopcl Resource Updates

You can dynamically change these sun3270x resources:

- Keyboard map
- Full IBM 16-color support
- Base color support
- Monochrome switches
- Uppercase only
- Dump screen output mechanism
- User keys

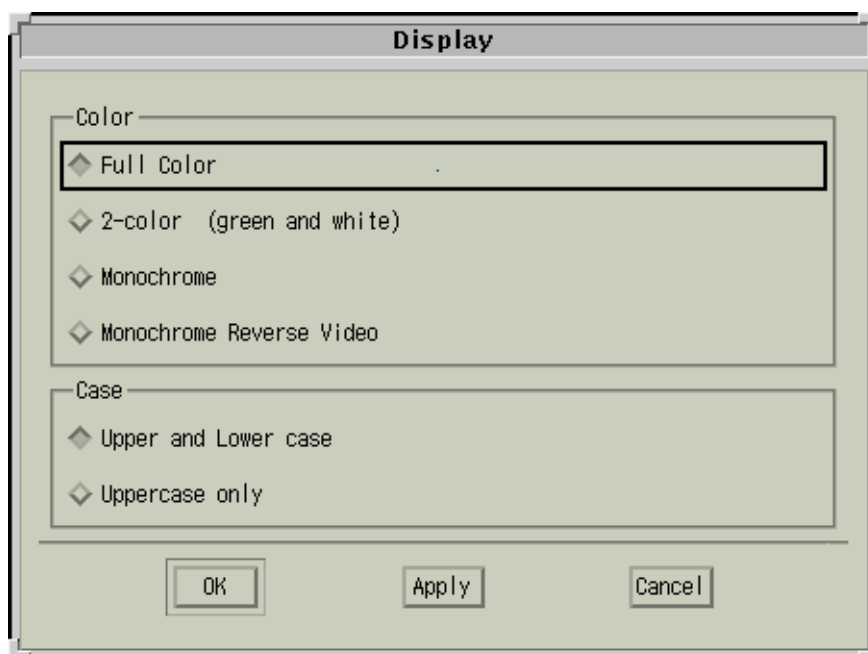
Use Settings under the Options button on the sun3270x Main menu.

Figure 4-1 shows the available settings.



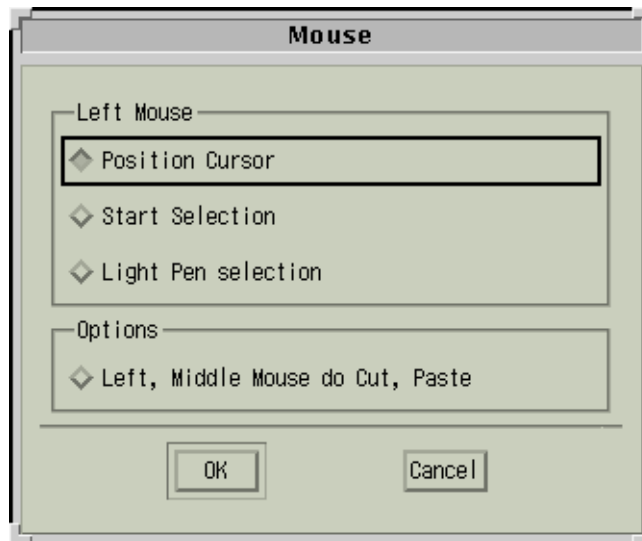
Figure 4-1 Options and Settings

1. **Keyboard** — Displays a layout of your keyboard with the IBM key values located on the keys of your keyboard. You can change your mapping simply by dragging an IBM key value to another key on your keyboard layout. Chapter 5, “Using Keyboard Mapper,” describes the keyboard mapper.
2. **Display** — Displays the Display dialog box in which you can customize the appearance of your screen display. The options are described below.



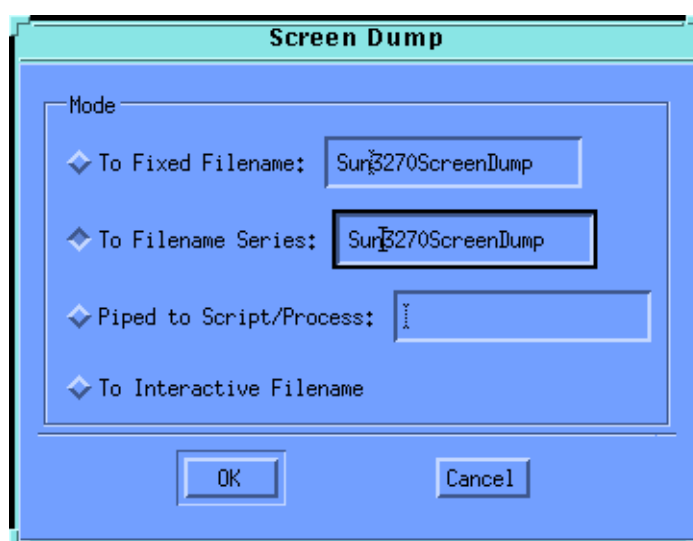
- **Color** — The following options are available:

- **Full Color** — Use full IBM 16 color mode
- **2-Color** — Use Base Color mode
- **Monochrome** — Use black and white display
- **Monochrome Reverse Video** — Use reverse video display (invert foreground and background colors).
- **Case** — The following options are available:
 - **Upper and Lower case** — Use mixed case.
 - **Uppercase only** — Use only uppercase characters on the display.
- 3. **Mouse** — Displays the Mouse dialog box in which you can change the function of the mouse buttons. The options are described below.



- **Left Mouse** — The left mouse button can have the following functions:
 - **Position Cursor** — Moves the cursor to the selected position.
 - **Start Selection** — Start a new edit selection without moving the cursor.
 - **Light Pen selection** — Enables the mouse to function as a Selector Light Pen. See Section 3.11, “Special Functions” in Chapter 3, “Using sun3270x,” for a detailed description of the Light Pen.
- **Options** — Enables you to use the left mouse button to cut/copy and the middle mouse button to paste.

- **Screen Dump** — Displays the Screen Dump dialog box in which you can choose to where you want to write screen output. The options are described below. To activate a Screen Dump request, press Action->Screen Dump.



- **To Fixed File Name** — Write to specified file name.
- **To Filename Series** — Write to specified base file name with numeric index suffix.
- **Piped to Script/Process** — Write output to the specified command.
- **To Interactive Filename** — Prompt for action each time dump screen function executed.
- **User Keys** — Displays the User Keys dialog box in which you define the function of menu items. User keys appear on the Action menu when the Key Button Title and Command fields are *not* NULL. The button title appears on the Action menu, and the command is executed when the menu item is selected. Selecting these buttons is the same as pressing the

appropriate user-defined function keys. The command may be any Unix command or pipeline. The EHLLAPI session short name is substituted into the command string wherever the special characters %f are found.

The dialog box is titled "User Keys". It contains a section labeled "Key Definitions" which displays a table with three columns: "Key", "Button Title", and "Command". The table lists seven entries: Usr1, Usr2, Usr3, Usr4, Usr5, Actu, and Dactu. The "Usr1" entry is highlighted with a black border. At the bottom of the dialog are "OK" and "Cancel" buttons.

Key	Button Title	Command
Usr1	Usr1	%f
Usr2	Usr2	%f
Usr3	Usr3	%f
Usr4	Usr4	%f
Usr5	Usr5	%f
Actu	Unit Activate	%f
Dactu	Unit Deactiva	%f

4.4 Message Library Updates

The text in the sun3270x resources, such as buttons and labels, can be changed to meet your requirements. Sun distributes the Sun Message Database (BMD) utility with sun3270x. The sun3270x resource text strings are stored as messages in the BMD.

To change the sun3270x resource text string, update the appropriate message entry in the B327_msgLib file.

See Appendix E, "Error Messages," for more information about displaying and modifying Sun error messages.

4.5 ASCII/EBCDIC Translations

To change the `SUNWopcl` ASCII/EBCDIC translations, use the `-a file_name` command-line argument or the `.asciiEbcDicTableFilename` resource.

The format of this file is: 512 white-space separated hexadecimal numbers. The '#' character indicates that the remaining part of the line is a comment.

The first 256 characters are used for the EBCDIC / ASCII translation. `SUNWopcl` programs determine the translation of an EBCDIC character by using the EBCDIC code point of the character as an index from the start of the first set of 256 characters.

The second 256 characters are used for the ASCII / EBCDIC translation. `SUNWopcl` programs determine the translation of an ASCII character by using the character's ASCII numeric code as an index from the start of the second set of 256 characters. Non-mapped characters in the ASCII / EBCDIC translation table should be mapped to the NULL character (0x00).

An example ASCII / EBCDIC translation file for Spanish follows in Code Example 4-2.

4.5.1 Sample ASCII/EBCDIC Translation Table

Code Example 4-2 Example ASCII/EBCDIC Translation Table

```
#
# Spanish Translation Tables
#

#
#           EBCDIC to ASCII
#
#

20 20 20 20 20 20 20 20 20 20 20 20 0a 0d 20 20 # 00-0f
20 20 20 20 20 20 0a 20 20 20 20 20 20 20 20 20 # 10-1f
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 # 20-2f
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 # 30-3f
20 20 20 20 20 E0 E1 20 20 E7 20 5B 2E 3C 28 2B 7C # 40-4f
20 E9 20 20 E8 ED 20 EF 20 26 5D 24 2A 29 3B AC # 50-5f
2D 2F 20 20 C0 C1 20 20 20 20 F1 2C 25 5F 3E 3F # 60-6f
20 C9 20 20 C8 CD 20 CE 20 60 3A BA D1 27 3D 22 # 70-7f
20 61 62 63 64 65 66 67 68 69 20 20 20 20 20 20 # 80-8f
20 6A 6B 6C 6D 6E 6F 70 71 72 AA 20 20 20 20 20 # 90-9f
20 7E 73 74 75 76 77 78 79 7A A1 BF 20 20 20 20 # a0-af
20 20 20 B7 20 20 20 20 20 20 5E 21 20 20 B4 20 # b0-bf
7B 41 42 43 44 45 46 47 48 49 20 20 20 F2 F3 20 # c0-cf
7D 4A 4B 4C 4D 4E 4F 50 51 52 20 20 FC 20 FA 20 # d0-df
5C 20 53 54 55 56 57 58 59 5A 20 20 20 D2 D3 20 # e0-ef
30 31 32 33 34 35 36 37 38 39 20 20 DC 20 DA 20 # f0-ff

#
# ASCII to EBCDIC
#
#

00 01 02 03 04 05 06 07 08 09 1e 0b 0c 0d 0e 0f # 00-0f
00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f # 10-1f
40 BB 7F 00 5B 6C 59 7D 4D 5D 5C 4E 6B 60 4B 61 # 20-2f
F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 7A 5E 4C 7E 6E 6F # 30-3f
7C C1 C2 C3 C4 C5 C6 C7 C8 C9 D1 D2 D3 D4 D5 D6 # 40-4f
D7 D8 D9 E2 E3 E4 E5 E6 E7 E8 E9 4A E0 5A BA 6D # 50-5f
79 81 82 83 84 85 86 87 88 89 91 92 93 94 95 96 # 60-6f
97 98 99 A2 A3 A4 A5 A6 A7 A8 A9 C0 4F D0 A1 00 # 70-7f
```

Code Example 4-2 Example ASCII/EBCDIC Translation Table (Continued)

00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	#	80-8f
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	#	90-9f
00	AA	00	00	00	00	00	00	00	00	9A	00	5F	00	00	00	00	#	a0-af
00	00	00	00	BE	00	00	B3	00	00	7B	00	00	00	00	AB	00	#	b0-bf
64	65	00	00	00	00	00	00	74	71	00	00	00	75	00	77	00	#	c0-cf
00	7C	ED	EE	00	00	00	00	00	00	FE	00	FC	00	00	00	00	#	d0-df
44	45	00	00	00	00	00	00	48	54	51	00	00	00	55	00	57	#	e0-ef
00	6A	CD	CE	00	00	00	00	00	00	DE	00	DC	00	00	00	00	#	f0-ff

Using Keyboard Mapper



Sun provides a graphical keyboard map utility with sun3270x to help you quickly determine how the corresponding IBM key values are associated with Sun keyboard functions. This graphical utility displays a layout of your keyboard with keys labeled with their corresponding IBM key values. It also allows you to dynamically update the current keyboard map while you are using the keyboard. IBM key values are simply picked up from one key and placed on another.

Consequently, the keyboard map utility provides a comprehensive help facility and the freedom to remap your IBM key values whenever you want.

5.1 Displaying the Keyboard Map

To invoke the keyboard mapper, select Settings->Keyboard.



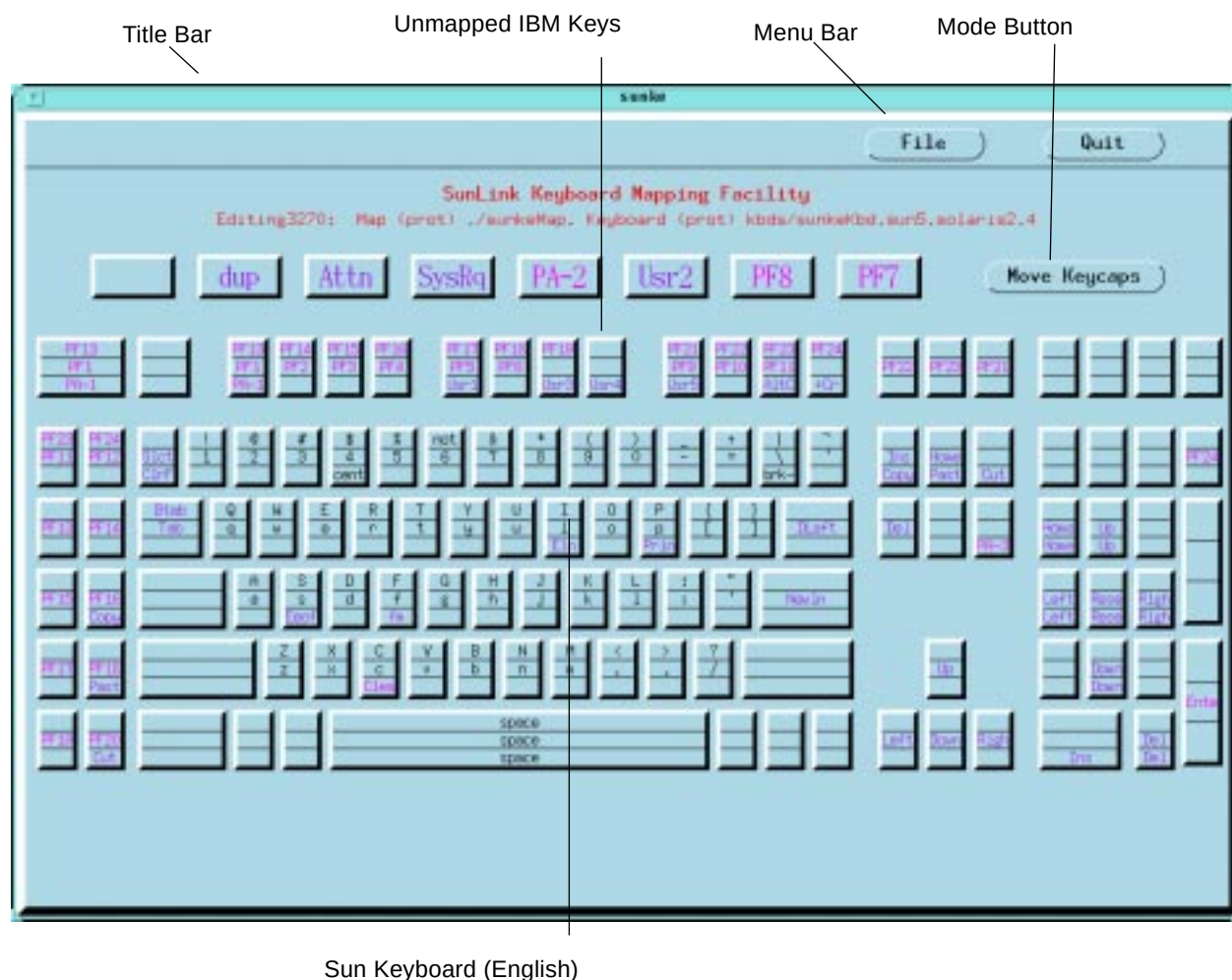
Figure 5-1 Options Keyboard Mapper

5.2 *Keyboard Map Layout*

Figure 5-2 illustrates an example sun3270x keyboard map layout. The keyboard map layout is made of five major components:

- Menu bar
- Title information
- Unmapped IBM keys
- Keyboard layout
- Mode Button

Many characteristics of the keyboard mapper can be customized, including: color, title, and keyboard layout (for new keyboards). See Appendix A, “Customizing the sun3270x Keyboard” for more information.



Sun Keyboard (English)

Figure 5-2 sun3270x Keyboard Map Layout

5.2.1 Menu Bar

The keyboard mapper provides a menu bar with three buttons:

Other Info (not supported)

Click the Other Info button to display a user-defined commentary about the keyboard map.

Next

The Next button is displayed if there is another display page for the keyboard layout. Click this button to view another page.

File

Click the File button to apply your updates to the keyboard map. The updates are stored in the file indicated in the "Editing:" message near the top of the keyboard mapper window. These files are always the ones currently used by the sun3270x; these are its keyboard map and keyboard layout files. See the `-k/.kbdMapFilename` in Chapter 3 to specify a keyboard map file name; the default file name is `sunkeMap`. See the `-Z /.kbdFilename` to specify a keyboard layout file name; the default file name is `sunkeKbd`. See the `-z / .kbdDirname` to specify a keyboard layout directory; the default is `./kbds`.

Note – You must have write permissions to the keyboard map file and the keyboard layout files to save your mapping updates. The keyboard mapper will *not* report the failure of a File operation.

Quit

Click the Quit button to exit the keyboard mapper. Be sure to select File before quitting if you want to save any mapping updates.

5.2.2 Title Bar

The title bar displays the name and version number of the keyboard mapper.

5.2.3 Unmapped IBM Keys

All unmapped IBM keys are stored in a “circular” array of keys. As you pick up IBM keys to map onto the keyboard layout, other unmapped IBM keys become visible. You can cycle through the array by moving the mouse over the unmapped IBM keys (but not the rightmost key unmapped key) and pressing the right mouse button.

5.2.4 Mode Button

The Mode button controls the overall functions of the keyboard mapper. Click the Mode button to cycle through the three possible keyboard mapper modes (see Table 5-1):

Table 5-1 Keyboard Mapper Modes

Mode	Cursor	Usage
Move KeyCaps	Top left arrow	Basic mode. Allows you to view and update the keyboard map.
Learn Kbd Keys	Pointing finger	Training mode. Enables you to “teach” the keyboard mapper the keyboard keys. See Appendix A, “Customizing the sun3270x Keyboard.”
Unlearn Keys	Skull and bones	Deprogramming mode. Enables you to tell the keyboard mapper to forget a keyboard key it previously taught. See Appendix A, “Customizing the sun3270x Keyboard.”

5.2.5 Single Key Layout

Each key in the keyboard layout is divided into three parts:

- Top — Shifted state of the key, such as 'A' and '\$'
- Middle — Normal state of the key, such as 'a' and '4'
- Bottom — Control state of key, such as CTRL-A and CTRL-Z

Figure 5-3 illustrates the different key states. The key on the left has been learned by the keyboard mapper (it is raised) and can be mapped to IBM key values. The key on the right has not learned (it is pressed) and cannot be mapped to IBM key values.

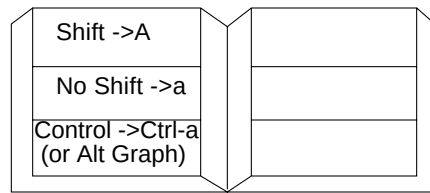


Figure 5-3 Single Key Format

5.2.6 Keyboard Layout

The keyboard layout occupies the bottom portion of your keyboard map display window. This area presents the mapping of IBM key values to keys on your keyboard. You can change these mappings.

The keyboard map display can run as a “player piano.” As you strike keys, the keyboard mapper presses the associated key in the graphical keyboard representation. Use the “player piano” feature to orient yourself with the association of keys on the keyboard and keys in the window.

5.3 Updating the Keyboard Map

Moving IBM key values to different keys on your keyboard is very simple: pick it up and drop it off. The following section explains this task in more detail.

5.3.1 Moving an IBM Key

To move an IBM key value:

1. **Place the mouse pointer above the IBM key value you want move and press the left mouse button.**
The IBM key value becomes attached to the mouse pointer.
2. **Place the mouse pointer (with the attached IBM key value) above the target key and press the left mouse button to drop off the IBM key value.**

Figure 5-4 shows the PF10 IBM key value in the process of being mapped. It has just been picked up from the unmapped IBM keys values. This IBM key value can be mapped to any key on the keyboard layout.

Drag and Drop IBM Key

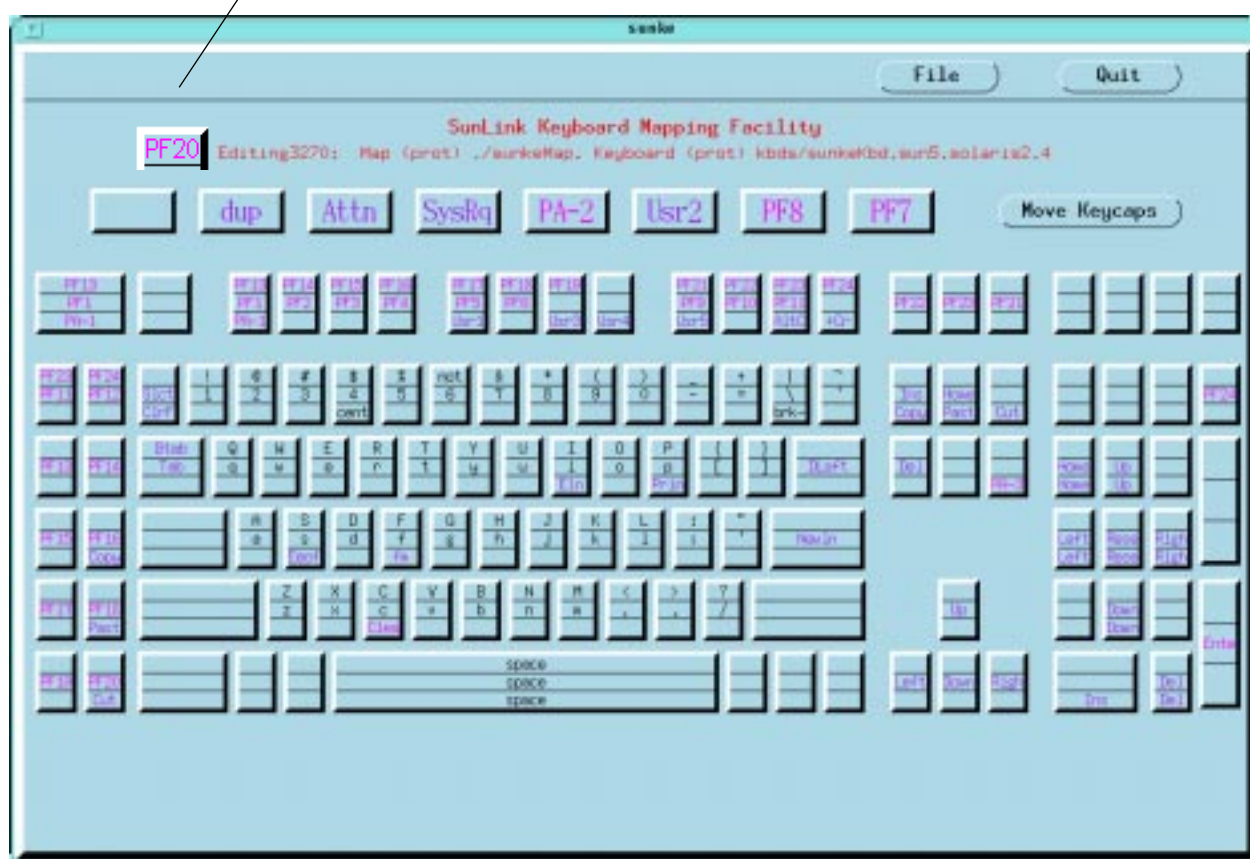


Figure 5-4 Previously Unmapped IBM Key Value (F20) Being Mapped

Similarly, Figure 5-5 demonstrates a change to a previous mapped IBM key value. In this case, the Back Tab (btab) IBM key value has been picked up and now can be deposited on another key.

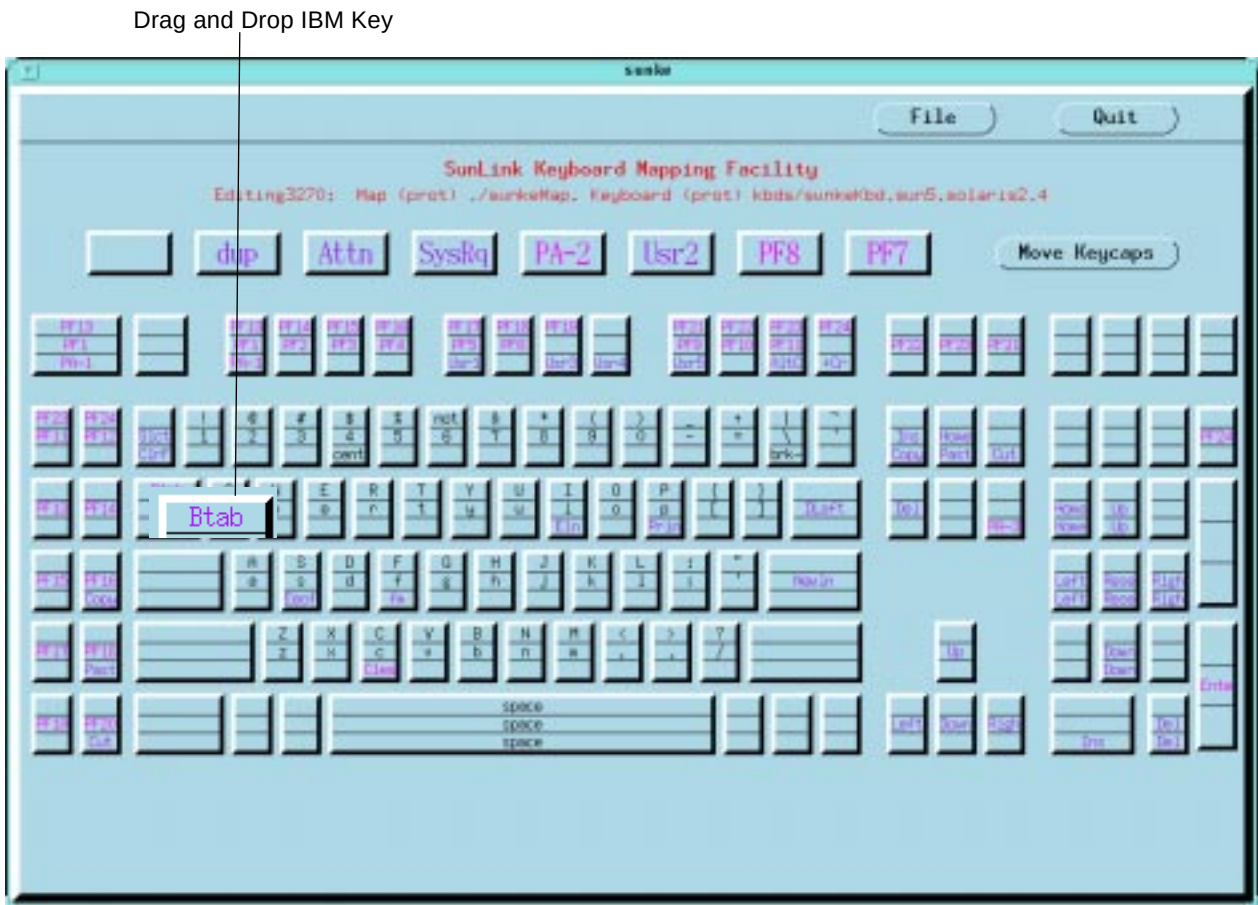


Figure 5-5 IBM Key Value (Btab) Being Re-mapped

If you drop an IBM key value onto a target key that has an IBM key value, the keyboard mapper swaps the two IBM key values—the incoming IBM key value is dropped off and the existing IBM key value is picked up.

5.3.2 Duplicating an IBM Key

- ♦ To duplicate an IBM key value, press the Shift key when you select the IBM key value.

5.3.3 Removing an IBM Key

- ♦ **To remove an IBM key value from the keyboard layout, pick up the IBM key value and drop it off on the unmapped IBM key values.**

5.4 Saving Keyboard Map Updates

To save keyboard map updates:

- 1. Click the File button.**

The File Confirmation menu is displayed.

- 2. Click the Ok button to save the files.**

The updates are stored in the files indicated in the Output Map File and Output Keyboard File fields of the File Confirmation menu. These fields are initialized from the “Editing:” message near the top of the keyboard mapper window.

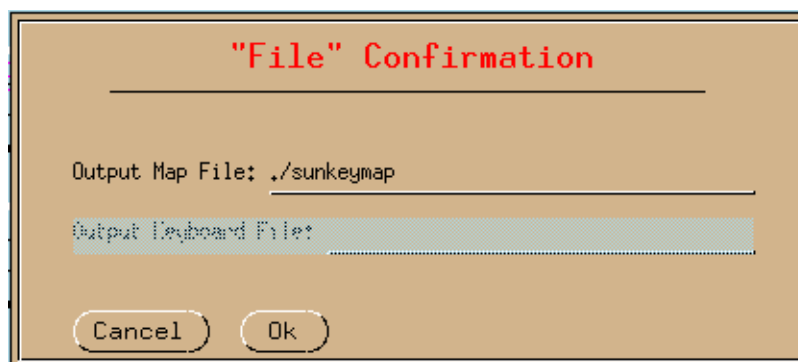


Figure 5-6 File Menu

5.5 Exiting Keyboard Map Display

- ♦ **To exit the keyboard mapper, click the Quit button.**

Using File Transfer



SUNWopc1 supports the exchange of binary, ASCII, and EBCDIC text files between the local Unix system and IBM host systems that support the IBM 3270 PC File Transfer (IND\$FILE) option. The sun3270x program provides an easy-to-use menu to facilitate file transfers.

Pacific Rim users: SUNWopc1 also supports the transfer of DBCS text files. If dbcsMode is turned on, the default for the File Transfer process is APVUFILE rather than (IND\$FILE).

See Appendix C, “Using PCFT,” for a description of the sun3270tty file transfer mechanism and for procedures for executing batch file transfers.

6.1 File Transfer Dependencies

You can transfer files to and from IBM host systems that support the IND\$FILE program within the following programs:

- CICS (Customer Information Control System), CICS / VS 5798-DQH, Version 1.00 or greater
- TSO (Time Sharing Option), MVS 5665-311, Version 1.00 or greater
- CMS (Conversational Monitoring System), VM / SP 5664-281, Version 1.00 or greater

Before you initiate a file transfer, you must be logged into one of the above-mentioned IBM host applications and positioned at the main prompt for that IBM host application. In TSO, for example, you should have the READY prompt displayed in your sun3270x window.

To execute a file transfer:

1. **Establish a communications link with the IBM host system.**
2. **Start sun3270x as an EHLLAPI server by specifying the `-e session_name` argument.**
3. **Log into the IBM host application TSO, CICS, or VM/CMS.**
4. **Ensure that the main prompt for the IBM host application is displayed.**

6.2 Starting a File Transfer

- ♦ **To start a file transfer, choose Action-->File Transfer from the sun3270x Main menu bar (Figure 6-1).**

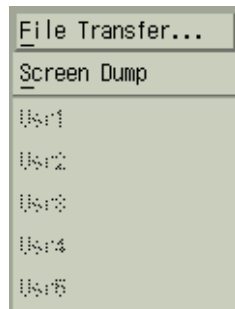
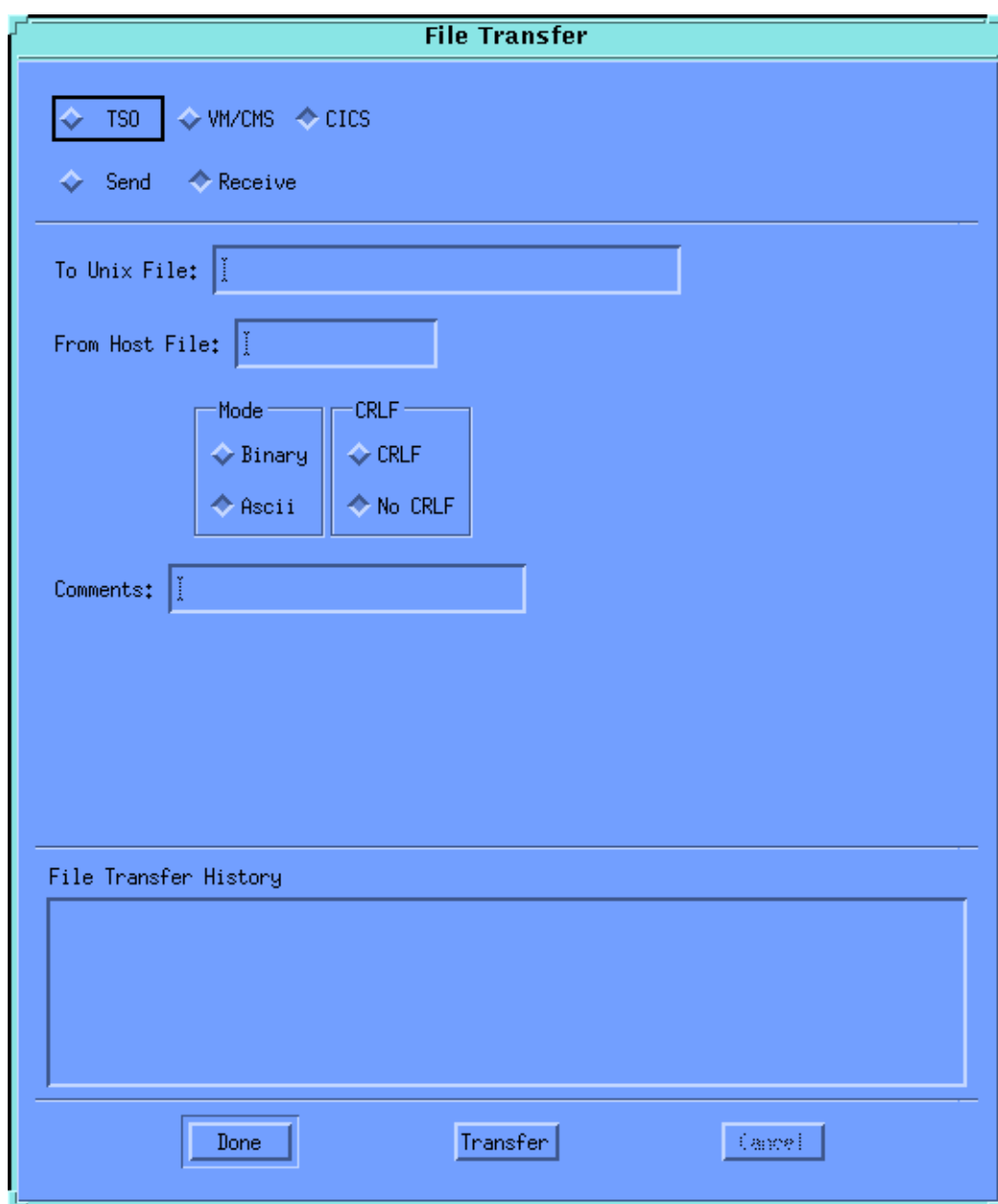


Figure 6-1 Action to File Transfer

Figure 6-2 contains the sample selections for the file transfer options. In this example, the readable file `/usr/abc` will be sent to the IBM host via the TSO application. The file will be stored in the `test1.abc.` data set.

Remember, before executing the file transfer, you must have the TSO READY prompt displayed on the sun3270x emulation screen.



The image shows a 'File Transfer' dialog box with a light blue background and a teal title bar. At the top, there are three radio buttons: 'TSO' (selected), 'VM/CMS', and 'CICS'. Below these are two more radio buttons: 'Send' and 'Receive'. A horizontal line separates the top section from the input fields. There are two text input fields: 'To Unix File:' and 'From Host File:'. Below these are two groups of radio buttons. The first group, labeled 'Mode', has 'Binary' and 'Ascii' options. The second group, labeled 'CRLF', has 'CRLF' and 'No CRLF' options. Below these is a 'Comments:' text input field. Another horizontal line separates the input section from the 'File Transfer History' section, which contains a large empty rectangular box. At the bottom, there are three buttons: 'Done', 'Transfer', and 'Cancel'.

File Transfer

☒ TSO ☐ VM/CMS ☐ CICS

☐ Send ☐ Receive

To Unix File:

From Host File:

Mode

☒ Binary
☐ Ascii

CRLF

☒ CRLF
☐ No CRLF

Comments:

File Transfer History

Figure 6-2 File Transfer

After setting the options described in the next section, click the Transfer button to begin the file.

- ♦ **To stop a file transfer in progress, click Done.**
- ♦ **To exit from the File Transfer utility, click Cancel.**

6.3 File Transfer Status

The running status of a file transfer is displayed at the bottom of the File Transfer, in the File Transfer History section of the screen. This section indicates:

- The start of the transfer
- A running tally of the bytes transferred
- The end of the transfer
- The status of the transfer

6.4 File Transfer Options

Before executing a file transfer, remember to move to the main READY prompt. Table 6-1 includes the file transfer options.

Table 6-1 File Transfer Options

Option	Function
IBM Host Application	
TSO	Logged into Time Sharing Option (TSO). If the IBM host system is running TSO-E, you must disable interrupts and exit from the session manager before beginning a file transfer. You can disable TSO interrupts with the term nobreak command. You can exit from the TSO session manager by clearing the screen and typing end.
VM/CMS	Logged into Conversational Monitoring System (VM/CMS).
CICS	Logged into Customer Information Control System (CICS).
Transfer Direction	
Send	Initiates a transfer from the local system to the IBM host system.
Receive	Initiates a transfer from the IBM host system to the local system.
UNIX Pathname	
UNIX File	Path name of file on the local system; required parameter.

Table 6-1 File Transfer Options (Continued)

Option	Function
IBM Host Filename	
Host File	File name on the IBM host; required parameter
TSO Data set Names	
<i>dataset_name</i>	A valid data set name contains 44 or fewer alphanumeric characters, in the form <i>user.dataset.name(member)</i> . Refer to the <i>IBM OS/VS2 Command Language Reference</i> manual for more information about TSO data set names and properties.
(<i>member_name</i>)	Name of member in the TSO data set specified by <i>host_file</i> . Optional positional parameter. Follows <i>host_file</i> immediately; no space separators.
password	Password used for TSO data set under password-protection. Optional positional parameter. Follows <i>host_file(member_name)</i> immediately; no space separators. TSO automatically prefixes the TSO data set name and member name with your TSO user ID. Use single quotation marks (') to avoid having the User ID prefixed to the TSO data set name and member name you specify.
CICS Data Set Names	
<i>dataset_name</i>	The CICS data set name can be up to 8 characters long. Refer to the <i>IBM Customer Information Control System (CICS/OS/VS) Version 1 Release 6 Installation and Operations Guide</i> manual for more information about CICS data set names and properties. Required parameter.
VM/CMS File Names	
<i>file_name</i>	The VM/CMS file name can be up to 8 characters. Refer to the <i>IBM Virtual Machine/System Product: CMS User's Guide</i> for more information about VM/CMS file names and properties. The format of the type in field is: <i>file_name.file_type.file_mode</i> .
<i>file_type</i>	VM/CMS file type. Required positional parameter
<i>file_mode</i>	VM/CMS file mode. Optional positional parameter. Default is 'A1'
Transfer Property	
create	Create the target file. File transfer cannot create a partitioned data set under TSO or CICS. It can create new members in an existing partitioned data set.
append	Append the contents of the transferred file to the target file. append cannot be used for members of a partitioned data set. The logical record length and record format values are the same as the original data set.
Transfer Mode	
ascii	Indicates that the transferred data is ASCII. Use this option, along with cr lf , for readable files. Do not use this option for binary files.

Table 6-1 File Transfer Options (Continued)

Option	Function
binary	Indicates that the transferred data is BINARY.
CRLF Option	
crlf	Indicates that carriage return/linefeed characters (CR/LF) act as record separators in the data transferred to the IBM host. On data transferred from the IBM host to the local system, the carriage return characters are removed. Use this option, along with ASCII text, for readable files. Do not use this option for binary files.
no_crlf	Indicates that carriage return/linefeed characters (CR/LF) do not act as record separators.
IBM Host File Attributes	
lrecl	Indicates the desired logical record length for the host data set (in bytes). Default: 80 bytes for new files; existing logical record length for existing files.
recfm	Indicates the desired record format for the host data set, where - f Specifies fixed-length records - v Specifies variable-length records - u Specifies undefined-length records. Default: Fixed-length records for new files (without <code>crlf</code> option specified); variable-length records for new files (with <code>crlf</code> option specified)
space(<i>prim</i> <, <i>sec</i> >)	Indicates the space allocation requirements for a new host data set, where <i>prim</i> is the primary and <i>sec</i> is the secondary allocation of blocks, tracks, and cylinders, where: <blocks(<i>x</i>)> is the smallest entity <tracks> is middle-sized entity <cylinders> is largest entity Default: default allocation value of <code>blksize</code> .
blksize	Indicates desired blocks size for the host data set (in bytes). Default: 80 bytes for new files; existing blocksize for existing files.

Using sun3287



sun3287 is a Sun IBM 3287 printer emulator process. The sun3287 program is a Unix application that you can invoke from the command line, a script, or as a menu option. Many sun3287 (and SUNWopc1) programs run in parallel; each sun3287 controls one session with an IBM host application.

sun3287 emulates both LU Type 1 (SCS) and LU Type 3 (DSC) printers. The sun3287 program dynamically switches between LU Type 1 and LU Type 3 emulation on receipt of the SNA BIND request. The SNA BIND request specifies the LU Type requested by the connecting application. While operating as an LU Type 1, the sun3287 program supports the SCS control codes described in Table 7-1.

Table 7-1 SCS Control Codes

Code	EBCDIC (Hex)	Name
BS	16	Backspace
BEL	2F	Bell function
CR	0D	Carriage return
ENP	14	Enable presentation
FF	0C	Form feed
GE	08	Graphic escape
HT	05	Horizontal tab
INP	24	Inhibit presentation
IRS	1E	Interchange record separator
LF	25	Line feed

Table 7-1 SCS Control Codes (Continued)

Code	EBCDIC (Hex)	Name
NL	15	New line
SA	28	Set attribute
SHF	2BC1	Set horizontal format
SLD	2BC6	Set line density
SVF	2BC2	Set vertical format
TRN	35	Transparent
VCS	04XX	Vertical channel select
VT	0b	Vertical tab

Use the `-h host_name` option to be attached to a Sun PU2.1 SNA server running on a different system in your local area network.

7.1 Dependencies

The `sun3287x` program attaches to the SunLink SNA 9.1 PU2.1 Server (`sunpu2.1`) using the `sun_pu2_espd` service name. This entry must exist in the `/etc/services` file. Normally, the installation of `SUNWpu21` makes this entry. If `sun3287` is running on a remote system, however, this service name must have the same port value on both the local and remote systems.

7.2 Starting sun3287

♦ To invoke `sun3287`, type the following command line:

```
sun3287 [-l lu_name] [-h host_name] [-p pu_name -n port-number]
        [-i] [-d] [-c command_line] [-s stream_command]
        [-f path_name] [-H horizontal_format] [-V vertical_format]
        [-a|A translation_name] [-P] [-t trace_flag] [-v]
```

The optional keywords on the command line are explained below:

-l *lu_name*

sun3287 requests that the LU services manager connect to the named LU (*lu_name*). The LU must be listed in the Sun PU2.1 SNA server configuration and the *lutype* parameter for the LU must equal 3. If the *lu_name* option is omitted, the first available LU is selected (active LUs are chosen first, inactive LUs are chosen next; only LUs with *lutype*=3 can be chosen).

-h *host_name*

sun3287 requests that the LU services manager connect to the Sun PU2.1 SNA server located on the named host. If the -h option is omitted, the local host is called.

-p *pu_name*

sun3287 requests that the SNA server assign it to an LU belonging to the named Physical Unit (PU). The PU must be configured in the local configuration file. If the *pu_name* option is omitted, the PU associated with the named or selected LU is used.

-n *port_number*

sun3287 requests that the LU Services Manager connect to the LU with the specified *port_number* on the Sun PU2.1 SNA Server (only if the *lutype* parameter for the LU is equal to 3). SunLink 9.1 port numbers begin at 2 and correspond to the LOCADDR of the LU directive in the SunLink 9.1 PU2.1 SNA Server configuration. This option can only be specified if the -p option is chosen. If the -n option is omitted and the -p option is specified, the first available LU on the specified PU is connected (active LUs are chosen first).

-i

Immediate mode of operation. If the requested LU is not available, sun3287 terminates.

-d

Becomes a daemon process during initialization. Detach from the controlling terminal and close *stdin*, *stdout*, and *stderr*.

-c *command_line*

Execute *command_line* after receiving each print stream. The name of the file in which the current print stream is stored can be represented in the *command_line* as %f. This symbol (%f) is expanded to the current path name of the print stream file and then the *command_line* is executed. The *command_line* is executed under the Bourne shell (sh). This option cannot be used with the -s option.

-s *stream_command*

Execute the *stream_command* as each print stream is received. With the option, no print *stream_file* is used for storage. Instead, the print stream data is piped directly to the *stream_command* as standard input (stdin). The popen(3) function is used.

-f *path_name*

Absolute or relative path name of the base-level file name used to store the print stream. Each new print stream is stored in a file whose name is the base-level file name with a unique number appended. For example, a base-level file name of abc would have associated print stream file named abc.0, abc.1, and so on.

-H *horizontal_format*

Defines the default horizontal format for the presentation space. This parameter is only valid while operating as an LU Type 1. The horizontal format is specified as follows:

MPP,LM,RM,TAB1,TAB2,...,TABn (See Table 7-2.)

Table 7-2 Horizontal Format Specifiers

Specifier	Function
MPP	The maximum presentation position, specifies the horizontal extent of the presentation surface. The default value is 132.

Table 7-2 Horizontal Format Specifiers

Specifier	Function
LM	The left margin specifies the column value of the left most presentation position. LM also serves as the first tab stop. This value must be less than or equal to the MPP. The default value is 1.
RM	The right margin specifies the column value of the right most presentation position. RM must be greater than or equal to LM and less than or equal to MPP. The default value is MPP.
TAB	Horizontal tab stops specifies the column values for the horizontal tab (HT) function. TABs must be greater than or equal to LM and less than or equal to RM. The maximum number of TABs that may be specified is 255. The default values are: 1, 4, 8, 16, 20, 24,..., 128, 132.

-V *vertical_format*

Defines the default vertical format for the presentation space. This parameter is only valid while operating as an LU Type 1. The vertical format is specified as follows:

MPL,TM,BM,TAB1,TAB2,...,TAB*n* (See Table 7-3.)

Table 7-3 Vertical Format Specifiers

Specifier	Function
MPL	The maximum presentation line specifies the vertical extent of the presentation surface. The default value is 66.
TM	The top margin specifies the line value of the top presentation line on the page. TM also serves as the first vertical tab stop. This value must be less than or equal to the MPL. The default value is 1.
BM	The bottom margin specifies the line value of the bottom presentation line on the page. BM must be greater than or equal to TM and less than or equal to MPL. The default value is MPL.
TAB	Vertical tab stops specifies the line values for the vertical tab (VT) function. TABs must be greater than or equal to TM and less than or equal to BM. TAB stop parameters 1 through 11 are also used to the line values for corresponding control channels 2 through 12; channel 1 is set by the TM. The maximum number of TABs that may be specified is 255. The default values are: 1, 8, 16, 24,..., 64.

-a *translation_name*

Identifies the file containing the ASCII/EBCDIC translation tables. See Section 4.5, “ASCII/EBCDIC Translations” for the file format.

-A *translation_name*

Identifies the file containing the ASCII/EBCDIC translation tables and requests that sun3287 print the tables during initialization. See Section 4.5, “ASCII/EBCDIC Translations” for the file format.

-P

Requests that sun3287 print the ASCII/EBCDIC translation tables during initialization. See Section 4.5, “ASCII/EBCDIC Translations” for the file format.

-t *trace_flag*

Trace mode of operation. The supplied value for the *trace_flag* is used to determine which internal traces are captured in the sun3287 trace log for debugging.

The bits in the *trace_flag* value indicate which trace to activate. The value of the *trace_flag* may be specified according to the syntax recognized by `strtol(3)`. The defined traces and associated *trace_flag* values are as follows:

0x0001 Trace buffers to/from SNA server API

0x0002 Trace buffers from the keyboard and to the screen

0x0004 Trace information on internal finite state transitions

Trace files are created in `/tmp` and named *sunlib_pid* and *sunlib_pid.1*. Traces accumulate in *sunlib_pid* until 1000 trace points have been logged. Then, this file is renamed *sunlib_pid.1* and tracing continues in *sunlib_pid*.

-v

Print the current release version number and terminate.

-w (or **-dbcs**)

These keywords enable Pacific Rim users to print dbcs.

After the sun3287 is started, it waits for applications to acquire it (through a BIND message) and transmit print data to it. sun3287 translates the print data from EBCDIC to ASCII before writing the print data to a disk file or stream.

7.3 Print Stream File Manipulation Examples

♦ **To move the print stream file to /home/new_file, type:**

```
% sun3287 -c "mv %f /home/new_file"
```

You can also pass the print stream to your own handling function (a Unix script or process) by entering a command like:

```
% sun3287 -c "my_print_job_script %f; rm %f"
```

♦ **To attach to an LU named BLU01104 (on the local system) and direct all print streams to a printer device named /dev/lp, type:**

```
% sun3287 -l BLU01104 -s "cat > /dev/lp" -H "132,10,122,18,24,56"
```

7.4 Starting suntn3287

To invoke suntn3287, type the following at the command line:

```
suntn3287 [-l lu_name] [-h hostname] [-n port_number] [-i][-d] \  
[-c command_line] [-s stream-command] [-f path_name] \  
[-H horizontal_format] [-V vertical_format] [-L line_length] \  
[-e eol] [-i] [-I term id] [-d] [-u] [-a/A translation_name] \  
[-P] [-t trace_flag] [-v]
```

Most of the suntn3287 command line keywords have the same meaning as their sun3287 counterparts with the following exceptions:

-I *terminal identifier*

suntn3287x negotiates terminal type sub-options as the specified terminal identifier.

If **-I** is omitted, the default is IBM-3287-1. This option is not normally used with an RFC-1647-compliant server, but is left in for backward compatibility. RFC-1647 servers only allow IBM-3287-1 for a printer terminal identifier.

-l *lu_name*

`suntn3270x` requests that the Logical Unit (LU) services manager connect to the named LU (*lu_name*). *lu_name* may either be a specific LU name or it may be a pool of LUs maintained by the TN3270 Server. This keyword is optional.

-p *pu_name*

This keyword is eliminated.

-d

Associate this printer instance with the display LU named with the **-l** *lu_name* option.

-w (or **-dbcs**)

These keywords enable Pacific Rim users to print dbcs.

For more information on Telnet, see Chapter 11, “Using `suntn3270x`.”

SunLink PU2.1 SNA Server Configuration



The `SUNWopc1` attaches to LU Type 2 ports (display terminals) on the SunLink 9.1 PU2.1 SNA Server. The `Sun3287` emulator attaches to LU Type 3 ports (printers) on the SunLink PU2.1 SNA server. Each port represents one LU. Ports for these SNA client applications must be configured in the local SunLink PU2.1 SNA server file.

Since LUs are SNA resources, you will need your LU ports defined in the SNA configuration. Chapter 8, “SunLink PU2.1 SNA Server Configuration,” discusses how to configure the Sun emulators in an SNA network.

8.1 SunLink PU2.1 SNA Server Configuration

The SunLink PU2.1 SNA server configuration file specifies the SNA resources associated with a Unix system. This configuration file is read during the SunLink PU2.1 SNA server (`sunpu2.1`) initialization. This chapter discusses the LU directive in the SunLink PU2.1 SNA server; the LU directive specifies SNA parameters for `SUNWopc1`. Refer to the *SunLink SNA PU2.1 9.1 SNA Server Configuration and Administration Manual* for more information on the SunLink PU2.1 SNA server configuration.

The configuration file is ASCII text. Each directive begins with a directive keyword and ends with a semicolon (;). The following conventions, shown in Table 8-1, are followed when describing each directive's syntax:

Table 8-1 Configuration Syntax

Mnemonic	Symbol	Description
Brackets	[]	Optional argument
Or-sign		Separates the options in a list
Uppercase	A	Must be entered as shown; can be either uppercase or lowercase
Lowercase	a	Variable argument, indicates the type of information
Commas	,	Must be entered as shown
Equal	=	Must be entered as shown
Single quote	'	Must be entered as shown
Slash slash	//	Indicates a comment; the rest of the line is ignored

There is no restriction on the use of blank lines, tabs, or spaces to separate directives and their arguments. Comments may appear anywhere and are preceded by “//”. Refer to Section 8.4, “Sample SNA Server Configuration” for an example configuration.

8.2 LU Directive

The LU directive defines the attributes associated with an LU port operating on the specified PU. Refer to Table 8-2. Absent parameters take default values.

Table 8-2 LU Port Attributes

	Attribute	Function
Syntax:	LU	NAME=lu_name, PU_NAME=associated_pu2, [LUTYPE=lu_type,] LOCADDR=n, [PACING=n];
Restrictions:	lu_name must be unique. associated_pu2 must be a PU2 NAME argument.	

Table 8-2 LU Port Attributes (Continued)

	Attribute	Function
Arguments:	NAME=lu_name	lu_name can be any alphanumeric string up to 8 characters. White space (tabs, spaces) is not allowed. NAME is used to identify this LU. You can specify an EHLLAPI session name for SUNWopcl programs by beginning the lu_name with HLA and ending the name with the EHLLAPI session name (e.g., HLAwxzyB specifies EHLLAPI session name B). Only valid for EHLLAPI session names A through H.
	PU_NAME=associated_pu2	associated_pu2 can be any alphanumeric string up to 8 characters. White space (tabs, spaces) is not allowed. PU_NAME is used to identify the PU2 directive associated with this station.
	LUTYPE=n	n is an integer specifying the type of LU (range 0-3). Default: 2. See Table 8-3.
	LOCADDR=n	n is an integer specifying the local address of this LU (range 1-254).
	PACING=n	n is an integer specifying the number of requests allowed in a pacing window. Default: 1.

Table 8-3 LU Types and Sun Client Programs

LU Type	Sun Client Program
0	SunLU0 API Program
1	Sun3770, SunLU0 API Program
2	SUNWopcl, SunLU0 API Program
3	sun3287, SunLU0 API Program. Always configured as LU Type 3, sun3287 emulates both LU Type 1 and LU Type 3, based on LU Type specified in received SNA BIND request.

8.3 Example Display Terminal Configuration

An SNA 3270 display terminal configuration example is given in Table 8-4.:

Table 8-4 SNA 3270 Display Configuration Example

```
LU    NAME=LU01,      // User defined name
      PU2_NAME=PU01,  // Name of PU2 this LU is on
      LUTYPE=2,       // LU Type 2 port (SUNWopcl port)
      LOCADDR=2,      // Local address of the LU
      PACING=1;       // Pacing window
```

An IBM SNA 3287 printer configuration example is given in Table 8-5:

Table 8-5 IBM SNA 3287 Printer Configuration Example

```
LU    NAME=LU01,      // User defined name
      PU2_NAME=PU01,  // Name of PU2 this LU is on
      LUTYPE=3,       // LU Type 3 port (Sun3287 port)
      LOCADDR=2,      // Local address of the LU
      PACING=1;       // Pacing window
```

8.4 Sample SNA Server Configuration

This section lists an example SunLink PU2.1 SNA server configuration file that can be used as a basis to configure LUs for SUNWopcl SNA clients. The sample configuration can be used with minimal changes. Normally, you only need to update the EOS directive and the PU2 ADDR argument. Refer to the SunLink PU2.1 server configuration documentation for details on the configuration parameters.

The sample configuration defines the following SNA connection shown in Figure 8-1.

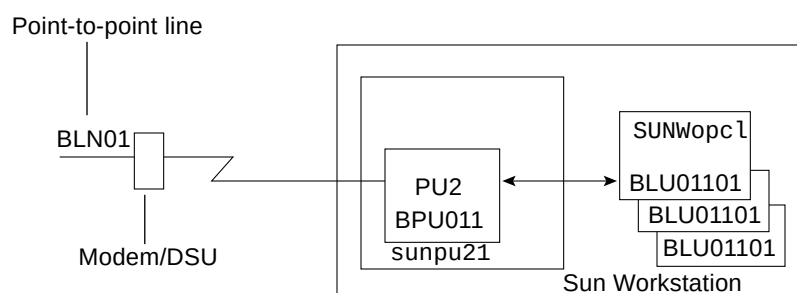


Figure 8-1 Point-to-Point Line Configuration Workstation

Code Example 8-1 SNA Server Sample Configuration

```
// Sun SunLink PU2.1 SNA Server Sample Configuration
//
// This sample configuration configures one point-to-point
// Line (BLN01); one Physical Unit (BPU011); and three Logical
// Units
// (BLU01101-BLU01103).
//
// The physical connection is realized via a 9600 bits/sec
// synchronous modem. The UNIX system is connected to the modem
// using serial port A (running the Sun synchronous driver,
// zbxdrv).
//
// You will need to configure the following parameters:
// - PU2 ADDR
//
// Remember:
//   SDLCLINE <--> VTAM LINE macro
//   PU2 <--> VTAM PU macro, where PUTYPE=2
//   LU <--> VTAM LU macro
//
```

Table 8-6 Server Configuration Parameters

Component	Parameters	Meaning
SDLCLINE	NAME=BLN01, Device='/dev/zbxa', CLOCK=external, DUPLEX=full, LINE=leased, NRZI=no, PAUSE=1, SPEED=9600;	// User defined name (8 char max) // UNIX device for this line (port A) // Clocking by modems (external source) // Full duplex line (point-to-point) // Point-to-point line // NRZ encoding // Wait for input during poll cycle (100 ms) // Line Speed (in bits/sec)
PU2	NAME=BPU011, LINK_NAME=BLN01, ADDR=x'c1', MAXDATA=265, OUTFRAME=7, MODULO=8, ACTIVITY_TIMEOUT=0, RETRIES=10;	// User defined name (8 char max) // Line name this station is on // Station address // Max data size for frame on link // Max outstanding frames allowed // SDLC is modulo 8 // Host activity timer -- none (in sec units) // Max retransmissions
Component	Parameters	Meaning
LU	NAME=BLU01101, PU_NAME=BPU011, LUTYPE=2, LOCADDR=2, PACING=1;	// User defined name (8 char max) // PU2 this LU belongs to // LU type 2 (3270) // Local address of LU // Pacing window
LU	NAME=BLU01102, PU_NAME=BPU011, LUTYPE=2, LOCADDR=3, PACING=1;	
LU	NAME=BLU01103, PU_NAME=BPU011, LUTYPE=2, LOCADDR=4, PACING=1;	

SNA Configuration for SUNWopc1



This chapter explains how to configure SUNWopc1 and sun3287 in the SNA host network configuration. Use the information in this chapter to help coordinate the updates to the local configuration and SNA host network configuration. The sample configuration, first described in Chapter 2, “Getting Started with SUNWopc1,” describes how to install SUNWopc1.

This chapter does *not* discuss all the requirements for extending the SNA host network configuration. Instead, it explains the additions needed to correctly configure SUNWopc1 in an SNA network. See the *SunLink 9.1 SNA PU2.1 Server Configuration and Administration Manual* for more information on configuring the local line and Physical Unit resources (including SDLC, Token Ring, and QLLC support).

Refer to the *IBM VTAM Installation and Resource Definition* for a complete description of SNA host network configuration.

9.1 SNA Configuration

An SNA network is a hierarchical network where many end-point devices communicate with a few host systems. To add a new device to the SNA network, an SNA host system programmer must update the SNA host network configuration (NCP/VTAM GEN).

The NCP/VTAM GEN lists all the SNA resources connected to an SNA communications controller. Four macros define the resources associated with a PU2 device emulated by the SunLink 9.1 PU2.1 SNA server. The LU macro defines the characteristics of the `SUNWopcl` that is attached to the corresponding LU port on the SunLink 9.1 PU2.1 SNA Server (see Table 9-1).

Table 9-1 NCP/VTAM Macros

MACRO	Description
GROUP	Specifies certain common characteristics and functions for a group of links and devices.
LINER	Represents the physical line connecting to the PU2 device.
PU	Represents the Physical Unit, the PU2 device.
LU	Represents a Logical Unit attached to the PU2 device. An LU provides a port for SNA applications to gain access to an SNA network. <code>SUNWopcl</code> is an SNA application. <code>sun3287</code> is an SNA application.

The LU type is determined by parameters that are transferred on the SNA BIND message which are received when the session is established LOGMODE definitions for `SUNWopcl` and `sun3287` are included later in this chapter.

The sample configuration shown in Figure 9-1 connects to the SNA network with one point-to-point link, one Physical Unit Type 2 (PU2), and three `SUNWopcl` client SNA programs (LUs).

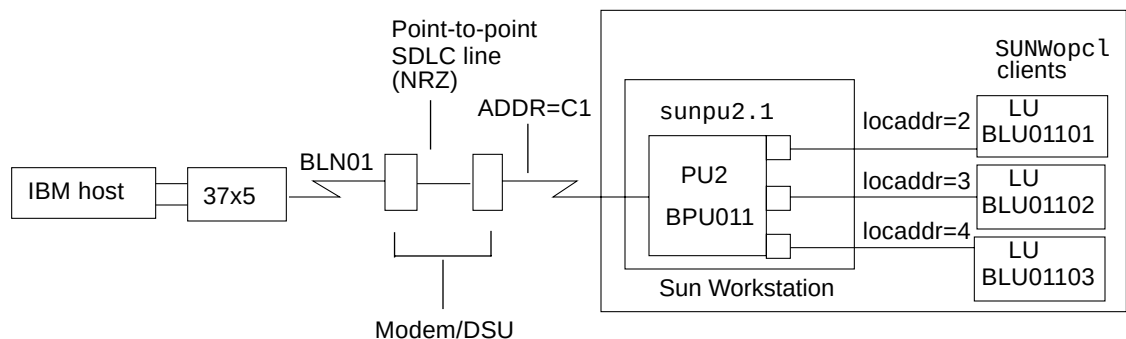


Figure 9-1 Sample SNA Configuration

The SunLink 9.1 PU2.1 SNA server emulates a 3274 Model C Series Cluster Controller (BPU011). Three SNA 3270 emulators (BLU01101, BLU01102, and BLU01103) are also present on the Sun Workstation. The Sun Workstation attaches to the SNA network through a Point-to-point SDLC line (BLN01). Part of VTAM/NCP GEN for this example is included in Code Example 9-1.

Note – This example does not include the configuration of a 3287 printer. To configure a 3287 printer, you must specify a 3287 logmode for an LU, for example, DLOGMOD = DCS2K.

Code Example 9-1 Partial VTAM/NCP GEN Example (1 of 2)

```

*****
*
* GROUP MACRO
*
*****
GRPSUN0          GROUP          DIAL=NO,
                                LNCTL=SDLC,
                                TYPE=NCP,
                                ISTATUS=ACTIVE,
*****
* LINE OPERANDS MOVED UP TO GROUP MACRO
*****
                                CLOCKING=EXT,
                                DISCNT=NO,
                                SERVLIM=5,
                                TRANSFER=9,
                                SPDSEL=NO,
*****
* PU OPERANDS MOVED UP TO GROUP MACRO
*****
                                IRETRY=YES,
                                MAXDATA=265,
                                MAXOUT=7,
                                PASSLIM=11,
*****
* LU OPERANDS MOVED UP TO GROUP MACRO
*****
                                MODETAB=ISTINCLM,
                                SSCPFM=USSSCS,
                                USSTAB=HIS3270,
                                PACING=1,
                                VPACING=1
*****
*
* LINE MACRO
*
*****
BLN01           LINE           ADDRESS=(01,FULL),
                                SPEED=9600,
                                NRZI=NO,

```

Code Example 9-1 Partial VTAM/NCP GEN Example (2 of 2)

```

                                DUPLEX=FULL
*****
*
* PU MACRO
*
*****
BPU011          PU          ADDR=C1,
                   PUTYPE=2
*****
*
* LU MACROS
*
*****
BLU01101        LU          LOCADDR=2, DLOGMOD=D4A32782
BLU01102        LU          LOCADDR=3, DLOGMOD=D4A32782
BLU01103        LU          LOCADDR=4, DLOGMOD=D4A32782

```

9.2 LOGMODE Tables

The VTAM LOGON Mode (LOGMODE) table requires entries to describe the operating characteristics of logical units (LUs). The association between the LU and its LOGMODE entry is created in the LU Macro of the VTAM/NCP GEN (or in the definition of a superior resource: GROUP, LINE, or PU).

A LOGMODE entry defines session and presentation values that are included in the session activation request (BIND) sent by the SNA host application to SUNWopcl or sun3287. The LOGMODE entry values specify the expected screen size, display formatting characteristics, maximum message sizes, and session protocol definitions.

SUNWopcl emulates IBM 3278 Model 1-5 display terminals, IBM 3279 Models 2A, 2B, 3A, 3B display terminals, and IBM 3287 printers (LU Type 3). The LOGMODE entry IBM distributes for these type of display terminals (e.g. D4A32782) can be used as a SUNWopcl LOGMODE entry.

Note – To execute file transfers with SUNWopcl, the first two bytes of the PSERVIC parameter must be: PSERVIC=X'0280...'.

The 02 defines the LU as an LU Type 2 device; the 80 indicates queries are supported.

Code Example 9-2 contains an example LOGMODE entry:

Code Example 9-2 Example LOGMODE Entry

```
LOG32782  MODEENT LOGMODE=LOG32782,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'90',
          COMPROT=X'3080',
          RUSIZES=X'85C7',
          PSNDPAC=0,
          SRCVPAC=0,
          SSNDPAC=0,
          PSERVIC=X'028000000000185000007E00'
```

Code Example 9-3 contains an example LU Type 1 LOGMODE entry:

Code Example 9-3 Example LU Type 1 LOGMODE Entry

```
DSC2K    MODEENT LOGMODE=DSC2K,
          FMPROF=X'03',
          TSPROF=X'03',
          PRIPROT=X'B1',
          SECPROT=X'90',
          COMPROT=X'3080',
          RUSIZES=X'8787',
          PSNDPAC=0,
          SRCVPAC=0,
          SSNDPAC=0,
          PSERVIC=X'010000000000185018507F00'
```

Code Example 9-4 contains an example LU Type 3 LOGMODE entry:

Code Example 9-4 Example LU Type 3 LOGMODE Entry

```
DSC2K  MODEENT LOGMODE=DSC2K,  
          FMPROF=X'03',  
          TSPROF=X'03',  
          PRIPROT=X'B1',  
          SECPROT=X'90',  
          COMPROT=X'3080',  
          RUSIZES=X'8787',  
          PSNDPAC=0,  
          SRCVPAC=0,  
          SSNDPAC=0,  
          PSERVIC=X'030000000000185018507F00'
```

9.3 Application Tables

Some IBM applications (for example, CICS) may require additional entries to application-specific tables that define the terminals and printers allowed to access the application and the type of allowed operations. See the appropriate application reference manual to prepare the host generation.

SUNWopcl Troubleshooting

10 

This chapter provides troubleshooting procedures for SUNWopcl. Sun provides several tools with SUNWopcl to assist in troubleshooting.

Note – Troubleshooting information for the SunLink 9.1 PU2.1 SNA Server and its SNA connections is documented in the *SunLink 9.1 SNA PU2.1 Server Configuration and Administration Manual*.

Use these utilities for troubleshooting problems:

- **Sun's Basic Message Database (BMD)**
This utility documents all errors, describes the cause, effect, and recovery action. Access the messages via the online database. The Online database utility and SUNWopcl messages are documented in Appendix E, "Error Messages."
- **SunLink 9.1 PU2.1 Operator (sunop)**
This utility displays status and statistical information for SunLink 9.1 PU2.1 SNA server resources. See the *SunLink 9.1 SNA PU2.1 Server Configuration and Administration Manual* for details on using sunop.
- **Logical data scope (sunscope)**
This utility is a data link level scope program that shows all transmitted and received frames on SNA data link connections. See the *SunLink 9.1 SNA PU2.1 Server Configuration and Administration Manual* for a guide to using sunscope.

- **Tracing:**

SUNWopc1 has extensive internal tracing facilities. Appendix D, “SunLink 3270 Tracing” lists a sample of the trace points captured by SUNWopc1. The SunLink PU2.1 SNA server also has extensive tracing capabilities.

10.1 Online Help

All error messages and display information are stored online in the Sun Basic Messages Database (BMD). You can view the message files or display information for specific messages by entering:

% bmsg COMPxxxx

where COMP represents the first four characters of the message group identifier, and xxxx is the message number. For example:

% bmsg B3270002

The information displayed about the message describes the meaning of the message and indicates the action you should take. See Appendix E, “Error Messages” for further information about the online help facility and a list of messages.

10.2 Operator Status Information

To determine the status of the SUNWopc1 session and to start sunop:

1. **Type the sunop command.**

You will see the Sun Controller message.

2. **Type the dis command.**

```
% sunop
Sun Controller
-> dis
(2) dis
->
OP200025 : (2) Link BLN01 - (2) Active
OP200020 : (2) Physical Unit BPU011 - (7) Active
OP20002a : (2) Logical Unit BLU01101 - (2) Active/Session
OP20002a : (2) Logical Unit BLU01102 - (2) Active/Attached
OP20002a : (2) Logical Unit BLU01103 - (2) Active
->
```

In this example, the first LU (BLU01101) is active and a SUNWopcl is in session with a host application. The second LU (BLU01102) is active and a SUNWopcl is attached to the LU but no session with a host application exists. In this state, SUNWopcl displays the logon banner. The third LU (BLU01103) is active but SUNWopcl is not attached to the LU port.

10.3 Logical Data Scope and Tracing

SUNWopcl processes can trace all messages that are sent and received. Messages are stored as trace points in the sunlib_pid file in /tmp.

- ♦ **To trace, start SUNWopcl with the trace option with “-t -1” where -1 specifies all trace points. See Appendix D, “SunLink 3270 Tracing” for an example SUNWopcl trace run. The SunLink 9.1 PU2.1 SNA server also has extensive tracing options.**

Use sunscope to monitor traffic on an SDLC line or an IBM Token Ring network.

- ♦ **To monitor an SDLC line connected to the synchronous device zbx0, type:**

```
% sunscope -e -d /dev/zbx0
```

- ♦ **To monitor an IBM Token Ring connection, where the adapter device is tr0, type:**

```
% sunscope -t -e -d /dev/tr
```

See the *SunLink 9.1 SNA PU2.1 Server Configuration and Administration Manual* for information on using sunscope and interpreting its output.

10.4 Recovery Steps

Follow the steps listed below to recover from conditions that prevent you from entering data in the SUNWopc1 screen. Follow each step until you recover from the problem. The placement of the 3270 keys on your Unix keyboard is described in the keyboard mapping file, sun3270map.

1. Press the Reset key.

This 3270 key unlocks the keyboard so you can re-enter data.

2. Press the Attention key.

This 3270 key resets most SNA host applications, returning you to the main application menu.

3. Press the System Request key.

This 3270 key switches you to the SSCP-LU session and back again. Switch to the SSCP-LU session and log off from the application.

4. Press Control-C.

SUNWopc1 will terminate.

10.5 Common Problems

Table 10-1 summarizes the types of problems you may encounter running `SUNWopc1` along with probable causes and corrective actions.

Table 10-1 Common Problems and Solutions

Symptoms	Probable Cause	Corrective Action
Unable to find keyboard map file message	sunkeMap file not found.	Check if sunkeMap is in your search paths (PATH environment variable).
sun3270 status line Readiness state is disconnected.	Unable to connect to SunLink 9.1 PU2.1; SunLink 9.1 PU2.1 not up	Start SunLink 9.1 PU2.1.
sun3270 status line Readiness state is disconnected. (SunLink 9.1 PU2.1 runs on remote system.)	Unable to ping SunLink 9.1 PU2.1 system Unable to connect to SunLink 9.1 PU2.1, network not up.	Establish network connectivity.
	Unable to connect to SunLink 9.1 PU2.1, incorrect <i>hostname</i> specified.	Invoke sun3270 with <i>hostname</i> of the remote SunLink 9.1 PU2.1 system. Use <code>-h hostname</code> .
	Unable to connect to SunLink 9.1 PU2.1, services for <code>brxadmin_pu2</code> and <code>brx_pu2_espd</code> do not match on both systems.	Ensure <code>/etc/services</code> files match on both systems. Ensure NIS has updated services.
sun3270 status line Readiness state is disconnected. (sunop shows LU active.)	Another application is using port, or, <code>lutype != 2</code> .	Check <code>lutype = 2</code> ; use another LU
sun3287 reports LU unavailable (sunop shows LU active.)	Another application is using port, or <code>lutype != 3</code> .	Check <code>lutype = 3</code> ; use another LU
sun3270 status line Readiness state is disconnected. (sunop dis shows LU Inactive.)	SNA Host has not activated the LU.	Request SNA system programmer to activate the LU.
sun3270 reports opening of EHLLAPI server.	Unable to find EHLLAPI service name.	Ensure that <code>/etc/services</code> contains <code>sunehllapiA</code> through <code>sunehllapiH</code> services.
Inverse video for highlighting.	TTY inverse mode not set properly.	Update <code>/etc/termcap</code> file entries for mode not set appropriately. Ensure (so and se) standout.
Using default map message.	sun3270map file not found.	Check search paths for sun3270map. Ensure TERM environment variable is defined in sun3270map.

Table 10-1 Common Problems and Solutions (Continued)

Symptoms	Probable Cause	Corrective Action
Logon failed: 0821 sense code received	Bad session parameters.	Use LOGMODE D4A32782 (or similar)
sun3270 status line Mode state is Wait.	Improper operation executed.	See previous section for recovery steps.
Error from move(x, y)	Screen (window) sized too small.	Resize screen and try again.
Keyboard map layout is not displayed.	sunke program cannot be found.	Ensure sunke is in your search paths.
Keyboard map layout is not updated.	sun3270 user does not have proper permissions.	Ensure that sun3270 user has write premissions for sunkeMap.
File Transfer option inactive	No EHLLAPI session name.	Specify an EHLLAPI name when starting sun3270 (-e option).
File Transfer cannot be started.	pcft program cannot be found.	Ensure pcft is in your search paths.
pcft reports its failure to connect to the EHLLAPI server.	No EHLLAPI session name, or EHLLAPI session in use.	Specify an EHLLAPI name when starting sun3270 (-e option). Specify correct EHLLAPI session name for pcft (prefix one character name to host file name).
File Transfer does not transfers data; IND\$FILE command sent, no other activity	Query bit not set in LOGMODE for LU.	Ensure PSERVIC parameter of LUs LOGMODE has first two bytes equal to X'0280 ... '

Using suntn3270x

11 

This chapter describes how to start and run Sun's X-based Telnet 3270 emulator (suntn3270x). The suntn3270x program provides Telnet 3270 emulation for any system supporting the X Window standard.

The suntn3270 product supports two graphical interfaces:

- X Windows uses suntn3270x on systems supporting OpenWindows, Motif, or X Window Manager (as on workstations, PCs, and X-terminals).
- TTY-type terminals use suntn3270tty on terminals like VT100s, VT200s, and Wyse.

See Chapter 13, “Using sun3270tty” for more information about running suntn3270 on TTY-type terminals.

You can invoke this application from the command line, from a script, or as a menu option. Many suntn3270x programs can run in parallel: each suntn3270x controls one display window and one session with an IBM host application.

Use the “-h *host_name*” option to attach to a Telnet server running on a different system in your local area network.

For more information on suntn3287, see “Starting suntn3287” in Chapter 7, “Using sun3287.”

11.1 suntn3270 Dependencies

The suntn3270x program accesses the keyboard map file during start up. The default keyboard map file name is `sunkeMap`. suntn3270x searches the directories listed in your `PATH` environment variable to find `sunkeMap`. If the search fails, suntn3270x announces that will use an internal default map.

If you want suntn3270x to support EHLLAPI programs, you must have an entry in the `/etc/services` file that identifies the service name for the EHLLAPI session. If suntn3270x is running on a remote system or a network name server (for example, NIS) is used, you may have to update the `/etc/services` file. EHLLAPI service names begin with `sunehllapi` and end with the EHLLAPI session name; therefore, the “A” EHLLAPI session service name is `sunehllapiA`. Since suntn3270x supports EHLLAPI session names “A” through “Z”, “0” through “9”, and “a” through “z”, up to 62 EHLLAPI programs can run on one system. To start suntn3270x as an EHLLAPI server, use the `-e` keyword.

To execute a file transfer, you must start suntn3270x as an EHLLAPI server.

11.2 suntn3270 Keywords

Keywords control the suntn3270x program. These keywords are the same as `sun3270x` (described in Chapter 3, “Using sun3270x,” except as noted below.

All the suntn3270x keywords have parallel entries in the `SUNWopc1` resources file. suntn3270x reads the `SUNWopc1` resources file during start up; you can update this file to customize suntn3270x (see Chapter 4, “Customizing sun3270x” for more information). The resource name for each keyword is under the keyword and is prefixed with a `'.'` (period).

The following keywords describe the TN3270 server attachment:

suntn3270x requests connection to the TN3270 server located on the named host. If the `-h` option is omitted, the local host is called.

`-h host_name`

`.hostName: host_name`

`suntn3270x` requests connection to the TN3270 server on the specified port. If the `-n` option is omitted, the default Telnet port (23) is called.

`-n port_number`
`.portNumber: port_number`

`suntn3270x` negotiates terminal type sub-option as the specified terminal identifier with the TN3270 server. If the `-I` option is omitted, the default is IBM-3279-3.

`-I terminal_identifier`
`.termId: terminal_identifier`

`suntn3270x` negotiates terminal type sub-option as the specified model number with the TN3270 server. Defined values are: 2, 3, 4, and 5. If the `-I` option is omitted, the default is the Terminal Identifier option value.

`-S model_number`
`.modelType: model_number`

`suntn3270x` negotiates terminal type sub-option as the specified extended mode with the TN3270 server. If the `-E` option is specified, the terminal type sub-option value is suffixed with `-E`.

`-E`
`.extendedMode: on|off`

11.2.1 Miscellaneous Keywords

When `suntn3270` settings are saved in a resource file, selecting this option applies the changed settings to the current operation.

`-saveImpliesApplyMode`
`.saveImpliesApplyMode`

Causes `suntn3270` to preserve the current layout of any resource files, and simply updates the changed resources. This is helpful for commented resource files.

`-mergeResourcefile`
`.mergeResourcefile`

The left mouse button controls the cursor position.

`-leftMouseCurPostionMode`
`.leftMouseCurPostionMode`

Allows you to cut and paste lines into the text at the current cursor position. The pasted text is inserted, moving the subsequent text over.

-leftMouseCurPostionMode
-leftMouseCurPostionMode

Toggles the insert key between text insertion and text overwrite.

-G
.toggleInsertM0de

11.2.2 Host Graphics Keywords

Chooses whether or not to enable GDDM support during this run of the suntn3270. If GDDM support is enabled, query replies to the host indicate GDDM and the GDDM datastream will be fully supported. If GDDM support is not enabled, query replies to the host indicate NO GDDM support and any GDDM datastream data will be silently ignored. The default is OFF.

-gddm
.gddmMode on | off

This flag is similar to -gddm/.gddmMode described above, except it causes the suntn3270 to allocate a separate set of 3 planes in pixel memory for the Graphic Layer. On some hardware this yields faster screen updates and refreshes in both the text layer and the graphics layer. It yields smoother, real-time updates of the graphics layer as GDDM data stream arrives.

This mode of GDDM is required for GDDM mix modes OR and XOR, for supporting 3-plane LPSs in graphics layer, and for the GDDM crosshair graphics cursor. If this mode is not enabled, mix modes OR and XO will act the same as mix mode OVERPAINT, Query responses to the host will indicate no support for 3-plane LPSs, any Load PS structured field having a color field of other than 0 will be rejected, and the GDDM crosshair graphics cursor will be unavailable (only the cross graphics cursor will be available).

If both the -gddm and -gddmDirect are given, -gddmDirect takes precedence. The default is OFF.

-gddmDirect
.gddmDirectMode on | off

The type of X-cursor to use for the cross-style GDDM graphics cursor. The default is crosshair.

```
-graphicCursor    cursor_name  
.graphicCursor    cursor_name
```

Enables support for the Load Programmed Symbols structured field during this run of the suntn3270 or not. If enabled, query replies to the host indicate support for Load PS and such structured fields will be processed when seen. If not enabled, query replies to the host indicate NO support and any Load PS structured fields seen will be ignored. This enables/disables Load PS for both the text layer and the graphics layer. The default is OFF.

```
-loadps  
.loadpsMode       on | off
```

Pathname of the bitmap file to read and use for GDDM standard fill pattern number *i*. Values for *i* are 1 - 14. The default is gddmPati.

Examples: gddmPat1, gddmPat5, gddmPat12, etc.

```
-gddmPati         file_name  
.gddmPati         file_name
```

Causes the suntn3270 to consider GDDM datastream coordinate value (0, 0) to be the upper-leftmost pixel in the graphics layer rather than considering (0, 0) to be the center of the graphics layer, which is the normal interpretation. A small number of hosts do consider datastream (0, 0) to be the upper-left pixel in the graphics layer, hence this switch. The default is OFF.

```
-gddmUpperLeft  
.gddmUpperLeftMode    on | off
```

Enables support of the WCC(Reset) bit (WCC bit 1). If enabled, the suntn3270 performs reset actions described in the *IBM 3270 Information Data Stream Programmer's Reference* when the WCC(Reset) bit is on in the WCC of an Erase_Write or Erase_Write_Alternate command. (This bit is ignored on Write and Erase_all_Unprotected commands.) If not enabled, WCC(Reset) bit is always ignored. The default is OFF.

```
-R  
.wccReset          on | off
```

Note – This will be enabled if gddm support is enabled.

When running the suntn3270 with GDDM support, it functions as an 8-color device.

11.3 Stopping suntn3270x

You can terminate the suntn3270x display in the following ways:

- Click the “Quit” button.
- Use the Title Bar Menu.
Most window managers provide a “quit” function in the menu associated with the window frame.
- Kill the suntn3270x process.
When killed, suntn3270x is terminated.

11.4 suntn3270 Status Line

The current status of the emulated 3270 display station is continuously displayed on the last line of the suntn3270x emulation screen (below the horizontal bar).

Figure 11-1 illustrates a status line that shows that the user is currently connected to the TN3270 server (connected); the host is busy processing (Wait); the display is in insert mode; the TN3270 server is located on host BigBlue with port number 23; the EHLLAPI session name is A; and the cursor is on row 10, column 4.

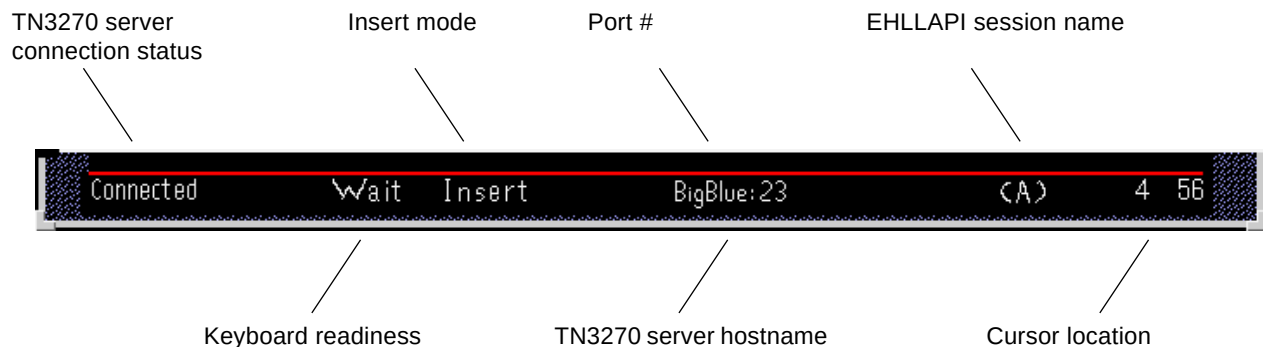


Figure 11-1 suntn3270 Status Line

The components of the status line are defined in Table 11-1.

Table 11-1 suntn3270 Status Line

Indication	Display
Status	The current screen owner or active session is displayed as follows:
	Connected The connection to the TN3270 server has been established.
	Disconnected The connection to the TN3270 server has not been established.
Readiness	The keyboard input capability is displayed as follows:
	Blank Keyboard input is enabled.
	Wait Keyboard input is disabled.
Modes	The terminal insert mode is displayed as follows:
	Blank The terminal emulator is not in insert mode.
	Insert The terminal emulator is in insert mode.
Host Name:Port	The current TN3270 server host name and port number are displayed. If suntn3270x supports an EHLLAPI session, the EHLLAPI session name is displayed in parenthesis after the port number.
Cursor	The current location of the cursor (row / column).

11.5 suntn3270 Examples

- ♦ To connect to a tn3270 server running on remote system BigBlue, and to specify terminal type sub-option IBM-3278-2-E, use the default Telnet port (23), serving as the EHLLAPI presentation space named A, and enter the following command:

```
% suntn3270x -h BigBlue -e A -I IBM-3278-2-E
```

or

```
% suntn3270x -h BigBlue -e A -S 2 -E
```


Using CG3270

12 

CG3270 adds the capability to support GDDM commands from the IBM host system. This enables the client 3270 product to emulate the IBM 3179G terminal in addition to the terminals already emulated. GDDM is only supported in the X versions of the 3270 and TN3270 emulators, `sun3270x` and `suntn3270x`. It is *not* supported by the TTY versions, `sun3270tty` and `suntn3270tty`.

CG3270 support is added by:

- Obtaining and installing the GDDM FlexLM licenses.
- Adding the following command line option(s) to the script that activates the SunLink Client 3270 binary.

These are descriptions of the switches and resources added to the 3270 emulator for supporting GDDM functions.

You have the option to enable or not to enable the GDDM support during this run of the emulator. If GDDM support is enabled, Query replies to the host to indicate that GDDM and the GDDM datastream is fully supported. If GDDM support is not enabled, Query replies to the host indicate NO GDDM support and any GDDM datastream data is ignored.

```
-gddm  
.gddmMode on | off
```

Default = off

This flag is similar to `-gddm/.gddmMode`, except that it causes the emulator to allocate a separate set of three planes in pixel memory for the graphic layer. On some hardware platforms this yields faster screen updates and refreshes in both the text layer and the graphics layer. It yields smoother, realtime updates of the graphics layer as GDDM datastream arrives. This mode of GDDM is required for GDDM mix modes of OR and XOR, for supporting 3-three plane LPS's in graphics layer, and for the GDDM "crosshair" graphics cursor. If this mode is not enabled, mix modes of OR and XOR are rendered the same as mix mode of OVERPAINT.

Query responses to the host indicate no support for three-plane LPSs. Any Load PS structured field with a color field other than 0 is rejected and the GDDM "crosshair" graphics cursor are unavailable (only the "cross" graphics cursor are available). If both the `-gddm` and `-gddmDirect` are given, the `-gddmDirect` takes precedence.

```
-gddmDirect
.gddmDirectMode on | off
```

Default = off

The attributes of the main display are changed with the geometry keyword described in Chapter 3, "Using sun3270x." When using the `gddm` option the geometry defaults to:

```
-geometry 750x550+100+100
```

The type of X-cursor to use for the "cross"-style GDDM graphics cursor.

```
-graphicCursor cursor_name
.graphicCursor cursor_name
```

Default = crosshair

Enable support for the "Load Programmed Symbols" structured field during this run of the emulator or not. If enabled, any query replies from the host indicate support for Load PS and such structured fields is processed when it is seen. If not enabled, Query replies to the host will indicate no support and any Load PS structured fields seen will be ignored. This enables or disables Load PS for both the text layer and the graphics layer.

```
-loadps
.loadpsMode on | off
```

Default = off

Pathname of the bitmap file to read and use for GDDM standard fill pattern number *i*, where *i* ranges from 1 through 14.

```
-gddmPati  file_name  
.gddmPati  file_name
```

Default = gddmPati

Examples: gddmPat1, gddmPat5, gddmPat12, etc.

It causes the emulator to consider GDDM datastream coordinate value (0, 0) to be the upper-leftmost pixel in the graphics layer and so forth, rather than considering (0, 0) to be the center of the graphics layer which is the normal interpretation. A small number of hosts do consider datastream (0, 0) to be the upper left pixel in the graphics layer, hence this switch.

```
-gddmUpperLeft  
.gddmUpperLeftModeon | off
```

Default = off

Enable support of the WCC(Reset) bit (WCC bit 1). If enabled, the emulator will perform reset actions. When the WCC(Reset) bit is on in the WCC of an Erase_Write or Erase_Write_Alternate command. (This bit is ignored on Write and Erase_all_Unprotected commands.) If not enabled, WCC(Reset) bit is always ignored.

```
-R  
.wccReset  on | off
```

Default = off. This will be enabled if gddm support is enabled.

12.1 Additional information.

Note – When running the emulator with GDDM support, CG3270 functions as an 8-color device.

To set the window size to 750 x 550, you need to specify the -geometry (.geometry) parameter. Example: “-geometry 750x550+100+100.”

Using sun3270tty

13 

This chapter describes how to start and run the Sun TTY-based SNA 3270 emulator, `sun3270tty`. The `sun3270tty` application emulates IBM 3278 display terminals (model 2) on terminals like VT100, VT200, and Wyse. It supports EHLLAPI programs and file transfer between your local system and IBM host systems.

The `sun3270tty` program is a Unix application that can be invoked from the command line, from a script, or as a menu option. Many `sun3270tty` programs can run in parallel: each `sun3270tty` controls one terminal and one session with an IBM host application.

- To view your current keyboard map, press the Help key sequence Escape-?.
- To Escape to shell, press the Escape key sequence, usually Control-x.
- To execute file transfers, see Appendix C, “Using PCFT.”
- To attach to a `sunpu2.1` SNA server running on a different system in your local area network, use the `-h host_name` option.

13.1 sun3270tty Dependencies

The sun3270tty program accesses the keyboard map file during start-up. The default keyboard map file name is sun3270map. The sun3270tty searches the directories listed in your PATH environment variable to find sun3270map. You can also use the SUNMAP3270 environment variable to specify the pathname of sun3270map or the -k command-line argument. If the search fails, sun3270tty announces that it will use an internal default map.

The sun3270tty program attaches to the sunpu2.1 SNA server (sunpu2) using the sun_pu2_espd service name. This entry must exist in the /etc/services file. Normally, the Sun configuration makes this entry if sun3270tty is running on a remote system. However, this service name must have the same port value on both the local and remote systems.

If you want sun3270tty to support EHLLAPI programs, you must make an entry in the /etc/services file that identifies the service name for the EHLLAPI session. Normally, the sun3270tty configuration makes this entry. If sun3270tty is running on a remote system or a network name server (NIS for example) is used, you may have to update the /etc/services file. EHLLAPI service names begin with sunehllapi and end with the EHLLAPI session name. Therefore, the "A" EHLLAPI session service name is sunehllapiA. Since sun3270tty supports EHLLAPI session names "A" through "H," up to eight EHLLAPI programs can run on one system.

- ♦ **To start sun3270tty as an EHLLAPI server, use the -e keyword or update the sunpu2.1 SNA server configuration.**

See Chapter 8, "SunLink PU2.1 SNA Server Configuration."

- ♦ **To execute a file transfer, start sun3270tty as an EHLLAPI server.**

13.1.1 sun3270tty Status Line

If the terminal display is less than 25 lines, sun3270tty only displays the status line (Operator Information Area, OIA) when the Status Line changes. Any subsequent screen changes cause the Status Line to disappear. You can check the Status Line by pressing the Status key sequence, usually Escape-o. The Status key sequence acts as a toggle; press it to display the Status Line, and press it again to hide the Status Line.

13.1.2 Intensified Support

For many terminal types, you can control the way the terminal displays intensified characters, such as reverse video, bold, and so on. Use the `/etc/termcap` file to control the type of intensity support for your terminal.

The `so` and `se` entries identify the character sequences to enter and exit “standout” mode (intensified). For a VT100 terminal, you can set the standout to bold mode by setting:

```
so = '\E[1m' and se = '\E[m'
```

13.1.3 sun3270tty Keywords

♦ **To invoke sun3270tty, type the following command line:**

```
sun3270tty [-e ehllapi_name] [-l lu_name] [-p pu_name -n port-number]
           [-h host_name] [-t trace_flag] [-v]
```

The following keywords describe the LU attachment:

`-a file_name`

Identifies the file containing the ASCII/EBCDIC translation tables to use. See Section 4.5, “ASCII/EBCDIC Translations” for the file format.

`-A file_name`

Identifies the file containing the ASCII/EBCDIC translation tables to use and requests that sun3270tty print the tables during initialization. See Section 4.5, “ASCII/EBCDIC Translations” for the file format.

`-c`

Override the default selection of Row 24 as the OIA if the terminal characteristics allow for it, for example if the device supports more than 24 lines).

`-d base_file_name`

Identifies the base file name to write screen output to when the dump screen option is selected. Each dump screen operation creates a new file with the `base_file_name` and a numeric suffix (incremented for each new operation). Default: sun3270ScreenDump.

The EHLLAPI session name and attributes are specified as:

-e *ehllapi_name*

sun3270tty takes *ehllapi_name* as the name EHLLAPI programs use to reference this **sun3270** display and keyboard buffers. The *ehllapi_name* must be one uppercase character between A and H. You can also specify the EHLLAPI session name in the **sunpu2.1** SNA server configuration. See Chapter 8, “SunLink PU2.1 SNA Server Configuration.”

-g

sun3270tty runs as a daemon process, with no display. Use this mode when you are “fronting” the 3270 display with your own application.

-h *host_name*

sun3270tty requests that the LU services manager connect to the **sunpu2.1** SNA server located on the named host. If the **-h** option is omitted, the local host is called.

-j

Enforce numeric editing on numeric fields. The default is no numeric editing. (See the description of resources record in Appendix B, “Mapping **sun3270tty** Keyboards” for a description of the **numStr** variable used to provide the user-specified numeric edit string.

The keyboard map file can be specified:

-k *key_map_pathname*

This is the pathname of the **sun3270map** keyboard map file. If the **-k** option is omitted, **sun3270map** searches the directories listed in your **PATH** environment variable for a file named **sunkeMap**.

-l *lu_name*

sun3270tty requests that the LU services manager connect to the named LU (*lu_name*). *lu_name* can be a specific LU name or a pool of LUs maintained by the **TN3270** server. This keyword is optional.

-n *port_number*

sun3270tty requests that the LU services manager connect to the LU with the specified *port_number* on the SunLink SNA PU2.1 SNA server (only if the **lutype** parameter for the LU is configured to equal 2). The *port_number* begins at 2 and correspond to the **LOCADDR** of the LU directive in the **sunpu2.1** SNA server configuration.

This option can only be specified if the `-p` option is chosen. If the `-n` option is omitted and the `-p` option is specified, the first available LU on the specified PU is connected (active LUs are chosen first).

`-p` *pu_name*

`sun3270tty` requests that the SNA server assign it to an LU belonging to the named Physical Unit (PU). The PU must be configured in the local configuration file. If the *pu_name* option is omitted, the PU associated with the selected LU is used.

Special functions include:

`-s`

Use only uppercase characters in the 3270 emulation window.

`-u`

Shows `sun3270tty` usage.

`-t` *trace_flag*

Trace mode of operation. The supplied value for the `trace_flag` is used to determine which internal traces are captured in the `sun3270tty` trace log used for debugging.

The bits in the `trace_flag` value indicate which trace to activate. The value of the `trace_flag` may be specified according to the syntax recognized by `strtol(3)`. The defined traces and associated `trace_flag` values are as follows:

- 0x0001 Trace buffers to/from SNA server API
- 0x0002 Trace buffers from the keyboard and to the screen
- 0x0004 Trace information on internal Finite State transitions.

Trace files are created in `/tmp` and named `sunlib_pid` and `sunlib_pid.1`. Traces accumulate in `sunlib_pid` until 1000 trace points have been logged. Then, this file is renamed `sunlib_pid.1` and tracing continues in `sunlib_pid`.

`-v`

Prints the current release version number and terminate.

13.1.4 sun3270tty Examples

- To attach to a sunpu2.1 SNA server on a remote system named bruno, type:

```
% sun3270tty -h bruno
```

- To attach to an LU named BLU01101 (on the local system) and serve as the EHLLAPI presentation space named A, type:

```
% sun3270tty -l BLU01101 -e A
```

- To start sun3270tty in an OpenLook window, use a shell script like the following:

```
#!/bin/sh
shelltool -Wt 9x15 -WG 765x390 -Wf 10 140 10 -Wb 0 0 0 \
-Wl "Sun3270" -default term.boldStyle Offset_X_and_Y_and_XY \
-default term.inverseStyle Same_as_bold sun3270tty \
$1 $2 $3 $4 $5 $6 $7 $8 $9 &
```

Customizing the sun3270x Keyboard



The sun3270x keyboard mapper is a Unix program named `sunke`. sun3270x starts `sunke` when you choose Keyboard Map from the Options menu. Chapter 5, “Using Keyboard Mapper,” discusses viewing and modifying the association between IBM key values and keys on your keyboard. This chapter describes how to change the “look” of the keyboard mapper and choose different colors, text, and keyboard layouts. Information in this appendix includes:

- Invoking `sunke`
- Customizing `sunke`
- Customizing `sunke` for a new keyboard

A.1 Starting sunke

You can start `sunke` program from a Unix command line. If you are creating a new keyboard, you will have to start and initialize `sunke` by specifying the `-g` option.

A.1.1 Files

The files used by `sunke` are as follows:

- `geometry` defines the physical layout of the keyboard.
- `keyboard` relates the `keysyms` generated by the X-server to a graphical key.
- `key caps` defines the logical keyboard functions.
- `mapping` relates each logical keyboard function to one or more X `keysyms`.

The `geometry` file is used to draw a picture of the keyboard on the screen. The picture, along with the `keysyms` generated by the X-server for each key, is saved in the `keyboard` file. The `key caps` file defines the 3270 logical keyboard functions and the EBCDIC code point corresponding to each function. The 3270 keyboard functions assigned to one or more X `keysyms` are saved in the `mapping` file.

The combination of `geometry` and X-server `keysyms` uniquely identifies the keyboard. This enables the SunLink 3270 emulation software to search a directory of `keyboard` files and select the correct one. It also enables a single `mapping` file to be projected on to any keyboard. When you modify your keyboard mapping, the modification is projected onto any keyboard you use because of the independence of the `mapping` file.

It is assumed that a system administrator has created the keyboard files for each keyboard and X-server pair employed. These files should be maintained in a central directory and restricted from update by everyone except the administrator. The administrator should also create an initial `mapping` file that should also be protected from update. Sun ships a `mapping` file and keyboard files for standard platforms with the SUNWopc1 product.

A.1.2 Dependencies

The `sunke` program accesses the keyboard map file and keyboard layout file during startup. The default keyboard map file name is `sunkeMap`. The default keyboard layout file name is `sunkeKbd`. `sunke` searches the directories listed in your `PATH` environment variable to find `sunkeMap` and `sunkeKbd`. Unless you specify the `-g` and `-c` options, `sunke` will exit if the keyboard map and keyboard layout files are not found.

A.2 Keywords

Keywords control the sunke program and define:

- New keyboard files
- Keyboard map file path name
- Keyboard layout file path name
- Window color scheme
- Window attributes
- Generic X keywords

All the sunke keywords have parallel entries in the sunke resources file. sunke reads the sunke resources file during startup. You can update this file to customize your sunke program. The resource name for each keyword is under the keyword and is prefixed with a '.' (period).

A.2.1 Keyboard Layout Definition Files

The following text explains different definition files and keyboard layout definitions.

- Path name of the file containing a keyboard layout definition. Default is none.

```
-g file_name  
.inGeometryFilename: file_name
```

- Path name of the file containing the IBM key values and their associated reference names. The format of this file is described in Section A.4, “Creating a New Keyboard Layout.” Default is none.

```
-c file_name  
.inKeycapsFilename: file_name
```

The following arguments control which keyboard map file and keyboard layout file to read on startup, and which files to update when the keyboard map or keyboard layout is changed:

- Path name of the file containing the keyboard map file. Default is sunkeMap in your current search paths.

```
-i file_name  
.inMapFilename: file_name
```

- Path name of the file containing the keyboard layout file. Default is `sunkeKbd` in your current search paths.
`-I file_name`
`.inKbdFilename: file_name`
- Path name of the keyboard file to create when changes are recorded for the current keyboard map. Default is to overwrite the current keyboard map file.
`-o file_name`
`.outMapFilename: file_name`
- Path name of the keyboard file to create when changes are recorded for the current keyboard layout. Default is to overwrite the current keyboard layout file.
`-O file_name`
`.outKbdFilename: file_name`

A.2.2 Window Attributes

The following text describes the window attributes related to this program.

- Width of the surrounding sides of a displayed key.
`sunke` attempts to create keys with the approximate width specified. Default is 6 pixels.
`-maxBevelWidth pixel_number`
`.maxBevelWidth: pixel_number`
- Size and position of the `sunke` display window.
The values are in pixel units. The form obeys the X geometry construct as defined by `XParseGeometry()`. Default is 900X750+100+100.
`-geometry widthxheight+Xposition+Yposition`
`.geometry: widthxheight+Xposition+Yposition`
- Path name of the file containing an icon for `sunke`. Default is Sun icon.
`-iconFilename file_name`
`.iconFilename: file_name`

- Title string for the X window Title Bar.
Default is the name of the sunke program.

 `-title title_name`
 `.title: title_name`
- String describing the keyboard mapper product and version of the keyboard map file.

 `-titleReadoutText text`
 `.titleReadoutText: text`
- X-cursor name when in “move” IBM key values mode.
Default is `top_left_arrow`.

 `-moveCursor cursor_name`
 `.moveCursor: cursor_name`
- X-cursor name when in Learn the Keyboard Layout mode.
Default is `hand2`.

 `-codeCursor cursor_name`
 `.codeCursor: cursor_name`
- X-cursor name when in Forget the Keyboard Layout mode.
Default is `pirate`.

 `-unicodeCursor cursor_name`
 `.unicodeCursor: cursor_name`
- Text for mode button when in move mode.
Default is `Move KeyCaps`.

 `-modeButtonMoveText text`
 `.modeButtonMoveText: text`
- Text for mode button when in move mode.
Default is `Learn Kbd Keys`.

 `-modeButtonCodeText text`
 `.modeButtonCodeText: text`
- Text for mode button when in move mode.
Default is `Unlearn Keys`.

 `-modeButtonUnicodeText text`
 `.modeButtonUnicodeText: text`

- Fonts `sunke` can use for display.
`sunke` chooses fonts based on the window size.

`-fontPool "font_name1 font_name2 ... font_nameN"`
`.fontPool: font_name1 font_name2 ... font_nameN`
- Text for the exit button.
Default is `Quit`.

`-exitButtonText text`
`.exitButtonText: text`
- Text for the file button
Default is `File`.

`-fileButtonText text`
`.fileButtonText: text`
- Text for the information button.
Default is `Other Info`.

`-infoButtonText text`
`.infoButtonText: text`

A.2.3 Window Colors

The following text describes the window color attributes. Colors are defined by their X color name or RGB weights as per the `XParserColor()` call.

- Color of the text for a “lifted” IBM key value.
Default is `black`.

`-capCursorTextColor color`
`.capCursorTextColor: color`
- Color of the “lifted” IBM key value.
Default is `red`.

`-capCursorBackgroundColor color`
`.capCursorBackgroundColor: color`
- Color of the window background.
Default is `light blue`.

`-background color`
`.background: color`

- Color of a button when selected.
Default is **gray**.

```
-xtsunButtonSelectedColor color  
.xtsunButtonSelectedColor: color
```
- Color of a button's text.
Default is **black**.

```
-xtsunButtonTextcolor color  
.xtsunButtonTextcolor: color
```
- Color of readout areas.
Default is **red**.

```
-xtsunReadoutTextColor color  
.xtsunReadoutTextColor: color
```
- Color of the text for a “lifted” IBM key value.
Default is **black**.

```
-capCursorTextColor color  
.capCursorTextColor: color
```
- Color of mapped IBM key values (characters).
Default is **black**.

```
-characterKeycapsColor color  
.characterKeycapsColor: color
```
- Color of mapped IBM key values (AID keys).
Default is **coral**.

```
-aidKeycapsColor color  
.aidKeycapsColor: color
```
- Color of mapped IBM key values (functions).
Default is **magenta**.

```
-miscKeycapsColor color  
.miscKeycapsColor: color
```
- Color of the illuminated side of a key.
Default is **white**.

```
-xtsunBevelLightColor color  
.xtsunBevelLightColor: color
```

- Color of the shadowed side of a key.
Default is **black**.

```
-xtsunBevelDarkColor color
.xtsunBevelDarkColor: color
```

A.2.4 Generic Arguments

Pass resource strings directly to the X-server.

```
-xrm resource_string
```

Identify the instance of **sunke** to the X-server. The *instance_name* is used to more specifically identify resource values in the **sunke** resources file that you want associated with this invocation **sunke**.

```
-name instance_name
.name: instance_name
```

Inform the X-server to display the **sunke** emulation window on a particular workstation console or X-term. **display_name** has the following format on Unix systems: *host_name:0.0*. Format of **display_name** follows **XOpenDisplay()**.

```
-display display_name
.display: display_name
```

General usage parameters for **sunke**.

```
-u
.useageMode: on|off
```

A.3 Customizing sunke

The **sunke** program has many attributes that you can change. In X Window environments, configurable attributes are called resources. The **sunke** resources belong to the **sunke** class.

When the **sunke** program starts, it searches for **sunke** resources in order of highest precedence:

- **./sunke**
- **\$XENVIRONMENT**; if it does not exist, **\$HOME/.Xdefaults-hostname**.
- Resource Manager property (**xrdb**). If it does not exist, **\$HOME/.Xdefaults**.

- `$HOME/sunke`
- `$XAPPLRESDIR/sunke`
- `/usr/lib/X11/app-defaults/sunke`

If a `sunke` resource is found, `sunke` initializes the defined resource with its specified value.

In the `sunke` resources file, a '*' (asterisk) preceding a resource indicates that all `sunke` programs should use that resource definition, such as
`*background: green.`

Alternatively, an instance name can be used to set a resource value only when the `-name` argument to the `sunke` program and the instance name match. Using an instance name makes it easy to customize the `sunke` program.

A resource specified with an instance name overrides any bindings using the '*' (asterisk) paradigm. `sunke` command line arguments override any settings established by `sunke` resource bindings.

Several `sunke` resources exist and fall into these categories:

- New keyboard files
- Keyboard map file path name
- Window color scheme
- Window attributes
- Generic X keywords

The `sunke` resources are described in Section A.1, "Starting `sunke`." All the `sunke` resources have parallel keyword definitions.

A.4 *Creating a New Keyboard Layout*

You will need to create a new keyboard layout if your keyboard is not a `sun4` or `sun5`-type keyboard for North American English. The procedure for creating a new keyboard is straightforward. Use the example keyboard layout files (`NAEnglishCaps` and `sun4Geom`) as your starting point. Most keyboards are similar to the layout described by the `sun4Geom` file. Similarly, most IBM key values are included in the `NAEnglishCaps` file.

To support a new keyboard, follow these steps:

1. Define an IBM key caps file.
2. Define a keyboard geometry file.
3. Teach `sunke` keyboard functions.
4. Map the IBM key caps to your new keyboard.

The following sections outline these steps. After defining the keyboard layout, you no longer need to specify the keyboard geometry or IBM key caps file — `sunke` stores away their information when you click the File button on the `sunke` display.

A.4.1 Defining an IBM Key Caps File

The IBM key caps file contains all the IBM key values that can be mapped to your keyboard. Start with the example file `NAEnglishCaps`. You can add values (or delete values) from this file. When `sunke` is started with the `-c` option to specify the new IBM key caps file, `sunke` displays the entries in this file in the un-mapped IBM key values section of the display (see Chapter 5, “Using Keyboard Mapper” for an illustration of `sunke` display).

IBM key values fall into four categories:

- 3270 AID keys
- 3270 function keys
- Sun special function keys
- EBCDIC characters

The format of the IBM key caps file is simple. Some entries of the form include:

Key Cap Name = Key Value

The Key Cap Name becomes the string associated with the IBM key value. Key Cap Names can be up to 5 characters. `sunke` displays the Key Cap Name; you move the Key Cap Name to keys in the keyboard layout to map the Key Cap Name's associated IBM key value to a keystroke.

For displayable characters, the Key Value is the character's EBCDIC code point (e.g., `A = 0xC1`). To support a UK English keyboard, for example, you need to add the EBCDIC £ value: `Pound = 5B`

For the 3270 AID keys, 3270 function keys, and the Sun special function keys, an associated string identifies the **Key Value**. Table A-1 contains a list of these key value strings (listed under mnemonics).

Table A-1 Keyboard Functions

Function	Mnemonic	Description
Cursor control:		
Backtab	backtab	Move cursor to previous field.
Cursor Down	down	Move cursor down one line.
Cursor Left	left	Move cursor left one character.
Cursor Right	right	Move cursor right one character.
Cursor Up	up	Move cursor up one line.
Home	home	Move cursor to first field.
New Line	newline	Move cursor to first field on next line.
Tab Forward	tab	Move cursor to next field.
Start/stop data entry:		
Attention	attn	Request for sending data.
Clear	clear	Clear screen.
Enter	enter	Initiate transfer of data to host.
Function Keys (PF1-24)	pf1-pf24	Initiate transfer of data to host.
Program Access (PA1-3)	pa1-pa3	Initiate transfer of data to host.
Reset	reset	Clear input inhibited status or insert mode.
Editing mode:		
Delete	delete	Delete character at cursor position.
Insert Mode	insert	Insert character at cursor position.
Field control:		
† Clear Field	clearfield	Clear field and move cursor to beginning of field.
† Delete Left	deleleft	Delete character to left of cursor.
DUP	dup	Duplicate operation for field.
Erase Field	eof	Erase to end of the current field.
Erase Input	ein	Erase all unprotected fields.
Field Mark	fm	Mark the end of a field or a subfield.

Table A-1 Keyboard Functions (Continued)

Function	Mnemonic	Description
Special function:		
Systems Request	sysreq	Switch to another session, such as LU-LU to SSCP-LU.
Cursor Select	cursel	Select field at cursor position.
Print	dumpscreen	Copy screen to file or program.
† Cut	cut	Cut selected portion of the screen.
† Copy	copy	Copy selected portion of the screen.
† Paste	paste	Paste selection to the cursor position.
† User defined function	usr <i>N</i>	Invoke user-defined function (N=1, 2, 3, 4, 5).

† Sun special keys

A.5 Sample IBM Key Caps File

Code Example A-1 Example Keycaps File (1 of 6)

```
; NAEnglishCaps
;
; %W% %G%
;
; =====
;
; This file contains example IBM key definitions (KeyCaps) for the sun3270x
; emulation program. This example file was used to construct the default
; keyboard map file: sunkeMap.
;
; The Sun keyboard mapper program (sunke) uses the NAEnglishCaps file
; to initially display the IBM key definitions -- the sun4Geom
; file defines the geometry of a sun4 keyboard. Once sunke produces a
; keyboard map file, the NAEnglishCaps file (and the sun4Geom) file are
; no longer needed: the information from these files are coalesced in
; the output keyboard map file (sunkeMap).
;
; The structure of the NAEnglishCaps file is simple:
;
;     KeyCap Name <== Key Value.
;
; For "displayable" characters, the Key Value is the characters EBCDIC
; code point. For special functions keys, an associated string identifies
; the Key Value.
;
; To invoke:
;
;     sunke -c NAEnglishCaps -g sun4Geom
;
; -----
;
;                               IBM Special Function Keys
;                               IBM KeyCap Name <== IBM Key Value
;
; -----
;
```

Code Example A-1 Example Keycaps File (2 of 6)

Enter	enter
Clear	clear
PA-1	pa1
PA-2	pa2
PA-3	pa3
PF1	pf1
PF2	pf2
PF3	pf3
PF4	pf4
PF5	pf5
PF6	pf6
PF7	pf7
PF8	pf8
PF9	pf9
PF10	pf10
PF11	pf11
PF12	pf12
PF13	pf13
PF14	pf14
PF15	pf15
PF16	pf16
PF17	pf17
PF18	pf18
PF19	pf19
PF20	pf20
PF21	pf21
PF22	pf22
PF23	pf23
PF24	pf24
Left	left
Right	right
Up	up
Down	down
Home	home
Reset	reset
Tab	tab
Btab	backtab

Code Example A-1 Example Keycaps File (3 of 6)

```
Ins                                insert

Del                                delete
Newln                             NewLine
Ein                               ein
Eeof                              eof
Attn                              attn
SysRq                             SysReq
dup                               dup
fm                                fm
CurSel                           fm

;-----;
;                                           ;
;                               Sun Special Function Keys           ;
;                               Name <== Value                      ;
;                                           ;
;-----;

Cut                                cut
Copy                              copy
Paste                             paste
ClrF                              ClearField
DLeft                             DeleteLeft
Print                             DumpScreen

;-----;
;                                           ;
;                               IBM Display Characters              ;
;                               Name <== Value (EBCDIC)            ;
;                                           ;
;-----;
;
```

Code Example A-1 Example Keycaps File (4 of 6)

space	40
a	81
b	82
c	83
d	84
e	85
f	86
g	87
h	88
i	89
j	91
k	92
l	93
m	94
n	95
o	96
p	97
q	98
r	99
s	a2
t	a3
u	a4
v	a5
w	a6
x	a7
y	a8
z	a9
A	c1
B	c2
C	c3
D	c4
E	c5
F	c6
G	c7
H	c8
I	c9
J	d1
K	d2

Code Example A-1 Example Keycaps File (5 of 6)

L	d3
M	d4
N	d5
O	d6
P	d7
Q	d8
R	d9
S	e2
T	e3
U	e4
V	e5
W	e6
X	e7
Y	e8
Z	e9
0	f0
1	f1
2	f2
3	f3
4	f4
5	f5
6	f6
7	f7
8	f8
9	f9
cent	4a
· — —	4b
<	4c
(	4d
+	4e
	4f
&	50
!	5a
\$	5b

Code Example A-1 Example Keycaps File (6 of 6)

```

*                                5c
)                                5d
;;                               5e
not                              5f

-                                60
/                                61
brk-|                            6a
,                                6b
%                                6c
_                                6d
>                                6e
?                                6f

`                                79
:                                7a
#                                7b
@                                7c
'                                7d
=                                7e
"                                7f

~                                a1
{                                c0
}                                d0
\                                e0

;-----
;
```

A.6 Defining a Keyboard Geometry File

For new keyboards, you may need to define a new keyboard layout (or geometry). The `sunke` accepts a simple text file to initialize the keyboard geometry. You specify the new geometry file by using the `-g` option when starting `sunke`.

`sunke` displays the keyboard layout in a window defined as 1000x1000 units. As you resize the `sunke` window, `sunke` readjusts the size and position of your keyboard geometry, always maintaining a 1000x1000 units inside the window (the units shrink or expand when you resize).

Each key in the geometry file is defined by:

- Position in the window:
 - X-coordinate (in units)
 - Y-coordinate (in units)
- Size of key:
 - Width (in units)
 - Height (in units)

Each line in the geometry file represents one key in the keyboard layout. The syntax is:

```
X-coordinate Y-coordinate Height Width
```

If one of these parameters is not specified, its value defaults to the parameter value of the line above. Positional values preceded by a '-' (hyphen) —are in absolute coordinates; otherwise, positional values are relative to the previous position.

For example, the entry

```
-15 -300 80 40
```

specifies a key in absolute position (15, 300) with a height of 80 units and a width of 40 units.

If the next entry is

```
45
```

then the next key is positioned 45 units greater in the X-coordinate (60, 300) and the size of the key is the same as the previous entry; the Y-coordinate, height, and width take the values of the previous entry.

A.6.1 Sample Keyboard Geometry File

Code Example A-2 Example Keyboard Geometry File (1 of 8)

```

;=====
;
;
; sun4Geom
;
; %W% %G%
;
;=====
;

;=====
;
;
; This file contains an example keyboard geometry for the sun3270x
; emulation program. This example geometry file was used to construct the
; default keyboard map file: sunkeMap.
;
; The Sun keyboard mapper program (sunke) uses the sun4Geom file
; to initially display the sun4 keyboard -- the NAEnglishCaps
; file defines the IBM keys. Once sunke produces a keyboard map file,
; the sun4Geom file (and the NAEnglishCaps) file are no longer needed:
; the information from these files are coalesced in the output keyboard
; map file (sunkeMap).
;
; The sunke program graphically displays the keyboard layout. The area within
; the window containing the keyboard is defined as 1000x1000 units.
;
; Each key in the keyboard window is defined by:
;
; -- Position:
;
; -- X-coordinate
; -- Y-coordinate
;
; -- Size
;
; -- Height
; -- Width
;

```

Code Example A-2 Example Keyboard Geometry File (2 of 8)

```
;
; The geometry file specifies a key by defining on one line the
; following:
;
;   X-coordinate   Y-coordinate Height Width
;
; If one of these parameters is not specified, its value defaults
; to parameter value of the line above. Positional values preceded by a '-'
; are in absolute coordinates; otherwise, positional values are relative
; to the previous position.
;
; For example, the entry
;
;   -15   -300    80    40
;
; specifies a key in absolute position (15, 300) with a height of 80
; units and a width of 40 units.
;
; If the next entry is
;
;   45
;
; then the next key is positioned 45 units greater in the X-coordinate
; (60, 300) and the size of the key is the same as the previous entry --
; the Y-coordinate, height, and width take the values of the previous
; entry.
;
; Note: the function buttons, the un-mapped IBM KeyCaps row, and
;       the display banner also are in the keyboard display window.
;
;
; To invoke:
;
;   sunke -c NAEnglishCaps -g sun4Geom
;
;=====
;
```

Code Example A-2 Example Keyboard Geometry File (3 of 8)

```

;-----;
;                                     ;
;               Function Key Row      ;
;               (top row of a sun4 keyboard) ;
;-----;
;
;-----;
; X      Y      Height Width ;
;-----;
-15 -300    80    40          ; L1
45                                     ; L2
-120                                     ; F1
45                                     ; F2
45                                     ; F3
45                                     ; F4
45                                     ; F5
45                                     ; F6
45                                     ; F7
45                                     ; F8
45                                     ; F9
45                                     ; F10
45                                     ; F11
45                                     ; F12
45                                     ; \
45      0      80    90          ; Delete
-815    0      80    40          ; R1
45                                     ; R2
45                                     ; R3
45                                     ; Num Lock

```

Code Example A-2 Example Keyboard Geometry File (4 of 8)

[illegible]

Code Example A-2 Example Keyboard Geometry File (5 of 8)

```

;-----
;
;
;               "QWRTY" Key Row
;               (third row of a sun4 keyboard)
;-----
;
;-----
; X      Y      Height Width
;-----
-15      -480      80      40      ; L5
45      ; L6
-120      0      80      60      ; Tab
65      0      80      40      ; Q
45      ; W
45      ; E
45      ; R
45      ; T
45      ; Y
45      ; U
45      ; I
45      ; O
45      ; P
45      ; [
45      ; ]
60      0      170      55      ; Return
-815      0      80      40      ; R7
45      ; R8
45      ; R9
45      0      170      40      ; Num Pad '+'

```

Code Example A-2 Example Keyboard Geometry File (6 of 8)

```

;-----;
;
;               "ASDFG" Key Row
;
;               (fourth row of a sun4 keyboard)
;-----;
;
;-----;
; X      Y      Height Width
;-----;
-15      -570     80     40      ; L7
45                ; L8
-120     0        80     75      ; Control
80       0        80     40      ; A
45                ; S
45                ; D
45                ; F
45                ; G
45                ; H
45                ; J
45                ; K
45                ; L
45                ;
45                ;
45                ; `
-815     0        80     40      ; R10
45                ; R11
45                ; R12

```

Code Example A-2 Example Keyboard Geometry File (7 of 8)

```

;-----;
;                                     ;
;               "ZXCVB" Key Row      ;
;               (fifth row of a sun4 keyboard) ;
;-----;
;
;-----;
; X      Y      Height Width ;
;-----;
-15      -660      80      40      ; L9
45                                     ; L10

-120      0      80      90      ; Shift
95      0      80      40      ; Z
45                                     ; X
45                                     ; C
45                                     ; V
45                                     ; B
45                                     ; N
45                                     ; M
45                                     ; ,
45                                     ; .
45                                     ; /

45      0      80      85      ; Shift

90      0      80      40      ; Line Feed
-815      0      80      40      ; R13
45                                     ; R14
45                                     ; R15
45      0      170      40      ; Enter

```

Customizing the sun3270x Keyboard

A.7 *Teaching sunke Keyboard Functions*

After you update the keyboard geometry and IBM key caps file, you are ready to train `sunke` about your keyboard keystrokes.

♦ **To train the keyboard, start `sunke` by typing:**

```
sunke -g geometry_file
```

`sunke` will start in teaching mode, indicated by the “hand” cursor. To train the keyboard, move the hand cursor above a pressed (unlearned) key and strike the similarly positioned key on your keyboard. `sunke` will beep and “raise” the key, indicating the association between the keystroke and that the keyboard layout has been learned.

Repeat the process for all the keys in your keyboard layout.

To retrain a key, move the hand cursor above the key, and strike a different key.

After training all of the keyboard keys, click the File option to save the keyboard file.

A.7.1 *Key Characterization*

The key characterization defines the association between the `keysym` and the virtual keycode the `SUNWopcl` interprets. `keysym` is the code sent by the X-server when a key is pressed. Discrepancies are known to exist within the `keysyms` generated by different X-servers. The key characterization menu enables you to modify the interpretation of a `keysym` received from the X-server. This is similar to performing an `xmodmap()` function within the `SUNWopcl` window only. This key characterization does not affect the actual `keysyms` generated by the X-server and has no impact on any other X-client application. The key characterization menu is shown in Figure A-1.

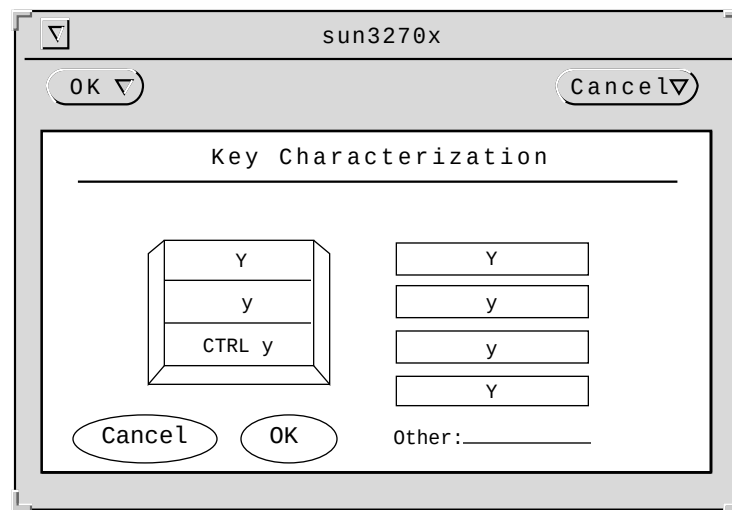


Figure A-1 Key Characterization Menu

To change the keysym associated with a slot on the key cap (Shift / normal / Control):

1. **Position the mouse on the slot and press the left mouse button.**
The text in the slot will be displayed in red.
2. **Click a keysym, generated by the X-server (one of the 4 slots on the right) by pressing the right mouse button on the desired selection, or type in any valid keysym name on the line labeled “Other.”**
3. **Press the return key. Once the selection is made, the keysym associated with the slot on the keycap is updated.**
4. **To apply the change, click the Ok button.**

Within the OpenWindows environment on Sun Workstations, the key labeled “print /sys req” produces the key characterization shown in Figure A-2. Within the Power Desktop environment on IBM RS-6000 systems, the key labeled “print /sys req” produces the key characterization shown in Figure A-3. The same keysym is used by these X-servers for different keystrokes. If the SUNWopcl is used in an environment using both of these

systems, a common keymap will invert the functions “print” and “sys req” on these systems. This situation can be avoided by using the “Key Characterization” to adjust the `keysym` interpretation from either of the X-servers.

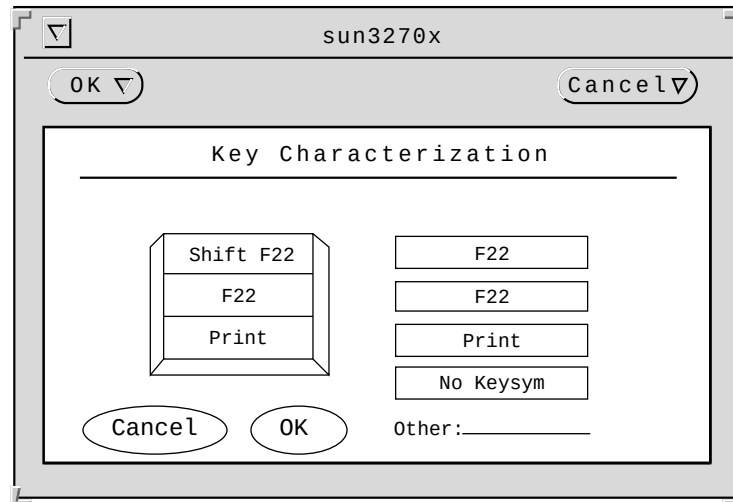


Figure A-2 sun3270x Print Key Characterization

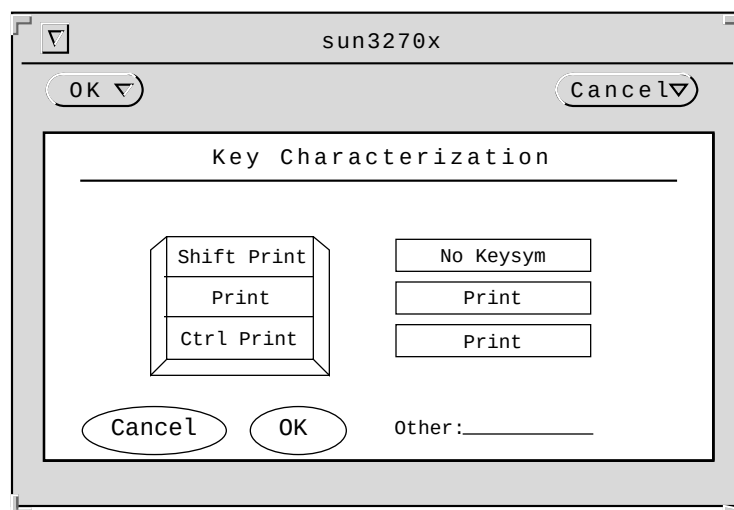


Figure A-3 IBM RS-6000 Print Key Characterization

A.8 Mapping IBM Key Caps

After training the keyboard, you are ready to map IBM key values to keys on your keyboard.

1. To map the keys, start sunke by typing:

```
sunke -c caps_file -I keyboard_file
```

- 2. Pick up IBM key values from the unmapped IBM key value list, and place them on keys in your keyboard layout.**
- 3. Toggle the Mode button (initially labeled Move KeyCaps) to the mapping mode — Move KeyCaps.**
- 4. Move the X-cursor over an IBM key value and press the left mouse button. Move the “picked up” IBM key value over a key on the keyboard layout.**
- 5. Press the left mouse button.**

Note – If you double click on the left mouse button, *sunke* picks up the next unmapped IBM key value from anywhere on the keyboard layout. This option saves time because you do not have to move back and forth between the keyboard layout and the unmapped IBM key values.

Repeat this procedure for all unmapped IBM key values.

See Chapter 5, “Using Keyboard Mapper,” for a complete description of mapping IBM key values.

Mapping sun3270tty Keyboards



Sun provides a keyboard map utility for sun3270tty. This utility helps you determine which keys affect IBM keyboard functions. You can also use this utility to change the mapping between keystrokes and IBM keyboard functions.

- ♦ **To display the current keyboard map when you are using sun3270tty, press the Help key sequence (usually Escape-?).**

The printout shows the IBM keyboard functions (IBM key values) and their corresponding input keystroke sequence.

You can modify the keyboard map. You can change key mappings or you can create key mappings for new keyboards. The name of the default keyboard map file is sun3270map.

The sun3270map file lists IBM key values by name, and associates these keys with an input character sequence for your device keyboard.

The SUNMAP3270 environment variable defines an absolute path name of the sun3270map file.

Your `.login` file specifies the `SUNMAP3270` environment variable.

♦ **To specify the environment variable in the `.login` file, type:**

```
setenv SUNMAP3270 /my_dir/my_map_file
```

This format allows you to specify a particular keyboard mapping file instead of the default file named `./sun3270map`.

♦ **To use the default mapping file format, type the following default format:**

```
No SUNMAP3270 defined
```

This format uses the default mapping file `./sun3270map`; the mapping file is translated each time `SUNWopc1` is started.

`sun3270tty` determines which keyboard map record to use by referencing your `TERM` environment variable. If the terminal entry is not found or the mapping file is not found, the default mapping is used.

Note – The `sun3270tty` keyboard map utility can interpret previous versions of the `sun3270map` file.

An example `sun3270map` file is distributed with the `SUNWopc1` emulator software. It must be located along the search path, for example, in a directory specified by the `PATH` environment variable) to be used.

B.1 Keyboard Map File Format

The keyboard map file has three major record types:

- `ibmCaps` record, which defines the IBM key values available for mapping.
- `aliases` record, which gives ASCII keystroke sequences more meaningful names.
- Terminal Map record, which maps the IBM key values to ASCII keystroke sequences.

All records are grouped with ‘{’ and ‘}’ (beginning and ending curly braces).

Comments start with a hash (#) and continue to the end of the current line. Comments may appear anywhere in the file.

B.1.1 ibmCaps Record

This record defines all the IBM key values that can be mapped to your keyboard. IBM key values fall into four categories:

- 3270 AID keys
- 3270 function keys
- Special Sun function keys
- EBCDIC characters

See Section 3.5, “Data Entry Overview” for a description of these different IBM key values.

The format of the IBM Key Caps file is:

Key Cap Name = key value

The Key Cap Name becomes the string associated the IBM Key Value. Key Cap Names can be up to seven characters.

For “displayable” characters, the key value is the character's EBCDIC code point (for example, A = 0xC1). To support a UK English keyboard, you need to add the EBCDIC £ value:

Pound = 5B

For the 3270 AID keys, 3270 function keys, and the special Sun function keys, an associated string identifies the key value. Table B-1 contains a list of these Key Value strings (listed under mnemonics).

Table B-1 sun3270tty Keyboard Functions

Function Name	Mnemonic	Description
Cursor Control:		
Backtab	backtab	Move cursor to previous field.
Cursor Down	down	Move cursor down one line.
Cursor Left	left	Move cursor left one position.
Cursor Right	right	Move cursor right one position.
Cursor Up	up	Move cursor up one line.
Home	home	Move cursor to first field.
New Line	newline	Move cursor to first field on next line.
Tab Forward	tab	Move cursor to next field.
Start/Stop Data Entry:		
Attention	attn	Request turn to send data.
Clear	clear	Clear screen.
Enter	enter	Initiate transfer of data to host.
Function Keys (PF1-24)	pf1-pf24	Initiate transfer of data to host.
Program Access (PA1-3)	pa1-pa3	Initiate transfer of data to host.
Reset	reset	Clear input inhibited status or insert mode.
Editing Mode:		
Delete	delete	Delete character at cursor position.
Insert Mode	insert	Insert characters at cursor position.
Field Control:		
†Clear Field	clearfield	Clear field and move cursor to start of field.
†Delete Left	deletelleft	Delete character to left of cursor.
DUP	dup	Duplicate operation for field.
Erase Field	eof	Erase to end of the current field.
Erase Input	ein	Erase all unprotected fields.
Field Mark	fm	Mark the end of a field or a subfield.

Table B-1 sun3270tty Keyboard Functions (Continued)

Function Name	Mnemonic	Description
Special Function:		
Systems Request	sysreq	Switch to other session (i.e. LU-LU to SSCP-LU).
Print	dumpscreen	Copy screen to file or program.

† Special Sun keys

B.1.2 aliases Record

Use this record to name ASCII character sequences so that their meaning is easily discernible when a SUNWopc1 user requests help. One `aliases` record can exist for each terminal map defined in the `sun3270map` file. The form of an `aliases` record entry is:

```
alias = ASCII character sequence
```

where, `alias` is up to seven characters, and `ASCII character sequence` represents an keyboard input data as defined below.

B.1.3 Terminal Map Record

This record contains the keyboard map (and possibly one `aliases` record) for one or more terminal types. Multiple terminal types can use the same Terminal Map record: list each terminal type, placing a '|' between terminal type names.

The format of a keyboard map entry is

```
Key Cap Name = ASCII character sequence or alias
```

where `Key Cap Name` is an IBM key value name, `ASCII character sequence` represents an keyboard input data as defined below, and `alias` is an alias name defined in the terminal type's `aliases` record.

B.1.4 ASCII Character Sequence

The ASCII input characters sequences are set-off by single quotation marks “” (single quotation). Multi-character sequences can only begin with an escape or control character. Special mnemonics exist for these characters (use '\ ' to escape the '\ ' and '^' characters):

- Escape \E
- Tab \t
- Newline \n
- Carriage Return \r
- Control ^

B.1.5 Sample sun3270map File

Code Example B-1 Sample sun3270map File (1 of 8)

```
#=====#
#
# sun3270map
#
# %W% %G%
#
#=====#

#=====#
#
# This file contains example keyboard maps for the sun3270tty
# emulation programs. This file is read during
# start-up and used to initialize the IBM keys of a 3270 display
# station keyboard to the keys of your keyboard (e.g. VT100, Wyse, etc.).
#
# This keyboard map file contains the following entries:
#
# -- ibmCaps record
#
# Associates names with the IBM key values of a 3270 display
# station keyboard. You can change the names if you wish.
#
# -- vt100 and vt200 keyboard map record
#
# Associates ASCII keystrokes with the IBM key values of a 3270
# display station keyboard. This record has two parts:
#
# -- aliases record
#
# Associates names with ASCII keystroke expansions. These
# aliases are especially useful for labeling keys which
# generate multi-character sequences when depressed (e.g.
# arrow keys and functions keys). You can change the
# aliases if you wish.
#
# -- ASCII keystrokes to IBM key value mappings
#
```

Code Example B-1 Sample sun3270map File (2 of 8)

```
#          Actual keyboard map for vt100 and vt200 keyboards. You
#          can change IBM key value to ASCII keystroke mappings.
#
#  -- sun and sun-cmd keyboard map record
#
#  -- aliases record
#
#          Associates names with ASCII keystroke expansions. These
#          aliases are especially useful for labeling keys which
#          generate multi-character sequences when depressed (e.g.
#          arrow keys and functions keys). You can change the
#          aliases if you wish.
#
#
#  -- ASCII keystrokes to IBM key value mappings.
#
#          Actual keyboard map for sun and sun-cmd keyboards. You
#          can change IBM key value to ASCII keystroke mappings.
#
#
# When sun3270tty starts, it references your local environment
# variable TERM to determine the terminal type. These programs then search
# the sun3270map to find matching the TERM type entry.
#
# These programs locate the sun3270map file by looking through your search
# paths as specified in your PATH environment variable.
#
# You can change and add to these entries to customize the keyboard map
# to your liking. For example, if you have a different terminal type,
# you will need to make another keyboard map entry for you new terminal
# type. Also, you can associate IBM key values to different keys
# on your keyboard -- maybe you prefer the IBM key Erase-to-end-of-field
# (eof) to be mapped to Escape Q (ESC-Q).
#
#
#=====#
```

Code Example B-1 Sample sun3270map File (3 of 8)

```
#=====#
#
#
#
#
#
#=====#

ibmCaps
{
    #
    # Non-display IBM keys (special functions and AID keys)
    #
    #

    Enter = enter      Clear = clear      SysReq = sysreq      PA-1 = pa1
    PA-2 = pa2          PA-3 = pa3          Left = left          Right = right
    Up = up             Down = down          Home = home          Reset = reset
    Tab = tab           Btab = btab          Insert = insert       Delete = delete
    Newline = nl        ErsInpt = einp        ErsEof = eof          Attn = attn
    Dup = dup           FldMark=fm

    PF1 = pf1           PF2 = pf2           PF3 = pf3           PF4 = pf4
    PF5 = pf5           PF6 = pf6           PF7 = pf7           PF8 = pf8
    PF9 = pf9           PF10 = pf10          PF11 = pf11          PF12 = pf12
    PF13 = pf13          PF14 = pf14          PF15 = pf15          PF16 = pf16
    PF17 = pf17          PF18 = pf18          PF19 = pf19          PF20 = pf20
    PF21 = pf21          PF22 = pf22          PF23 = pf23          PF24 = pf24

    #
    # Special Sun function keys
    #
    #

    Escape = escape      ClrFld = clrfld      Erase = erase         Help = help
    Status = status      Reshow = reshow      DmpScrn = dmpscr
```

Code Example B-1 Sample sun3270map File (4 of 8)

```
#
# Non-display and non-ASCII characters
#

space = 40          cent = 4a          not = 5f          'brk-|' = 6a

#
# Characters which need names because they are also interpretable
# sun3270map special characters
#
#

SemiCln = 5e        LBrace = c0          RBrace = d0          SQuote = 7d

#
# The rest ...
#
#

a = 81      b = 82      c = 83      d = 84      e = 85      f = 86      g = 87
h = 88      i = 89      j = 91      k = 92      l = 93      m = 94      n = 95
o = 96      p = 97      q = 98      r = 99      s = a2      t = a3      u = a4
v = a5      w = a6      x = a7      y = a8      z = a9

A = C1      B = C2      C = C3      D = C4      E = C5      F = c6      G = c7
H = c8      I = c9      J = d1      K = d2      L = d3      M = d4      N = d5
O = d6      P = d7      Q = d8      R = d9      S = e2      T = e3      U = e4
V = e5      W = e6      X = e7      Y = e8      Z = e9

0 = f0      1 = f1      2 = f2      3 = f3      4 = f4      5 = f5      6 = f6
7 = f7      8 = f8      9 = f9

. = 4b      < = 4c      ( = 4d      + = 4e      | = 4f      & = 50      ! = 5a
$ = 5b      * = 5c      ) = 5d      - = 60      / = 61      , = 6b      % = 6c
_ = 6d      > = 6e      ? = 6f      ` = 79      : = 7a      '#' = 7b      @ = 7c
= = 7e      " = 7f      ~ = a1      = e0

}
```

Code Example B-1 Sample sun3270map File (5 of 8)

```

=====#
#
#
#           Keyboard Map Record:
#           vt100 and vt200
#
#
#=====#

vt100 | vt200 | sun
{
    #
    # Aliases
    #
    #

    aliases
    {
        Enter = '^m'      Delete = '^?'      Return = '^m'      Tab = '^i'
        Up = '\E[A'       Down = '\E[B'       Right = '\E[C'     Left = '\E[D'
        BackSp = '^h'

        ESC-a = '\Ea'     ESC-b = '\Eb'     ESC-c = '\Ec'     ESC-d = '\Ed'
        ESC-e = '\Ee'     ESC-f = '\Ef'     ESC-g = '\Eg'     ESC-h = '\Eh'
        ESC-i = '\Ei'     ESC-j = '\Ej'     ESC-k = '\Ek'     ESC-l = '\El'
        ESC-m = '\Em'     ESC-n = '\En'     ESC-o = '\Eo'     ESC-p = '\Ep'
        ESC-q = '\Eq'     ESC-r = '\Er'     ESC-s = '\Es'     ESC-t = '\Et'
        ESC-u = '\Eu'     ESC-v = '\Ev'     ESC-w = '\Ew'     ESC-x = '\Ex'
        ESC-y = '\Ey'     ESC-z = '\Ez'

        ESC-1 = '\E1'     ESC-2 = '\E2'     ESC-3 = '\E3'     ESC-4 = '\E4'
        ESC-5 = '\E5'     ESC-6 = '\E6'     ESC-7 = '\E7'     ESC-8 = '\E8'
        ESC-9 = '\E9'     ESC-0 = '\E0'     ESC-- = '\E-'     'ESC-= ' = '\E='

        ESC-! = '\E!'     ESC-@ = '\E@'     'ESC-#' = '\E#'   ESC-$ = '\E$'
        ESC-% = '\E%'     ESC-^ = '\E^^'    ESC-& = '\E&'     ESC-* = '\E*'
        ESC-( = '\E('     ESC-) = '\E)'     ESC-_ = '\E_'     ESC-+ = '\E+'
        ESC-SP = '\E '    ESC-? = '\E?'
    }
}

```

Code Example B-1 Sample sun3270map File (6 of 8)

```

CTRL-B = '^b'    CTRL-D = '^d'    CTRL-L = '^l'    CTRL-N = '^n'
CTRL-U = '^u'    CTRL-X = '^x'    CTRL-Z = '^z'
CTRL-P1 = '^p1'  CTRL-P2 = '^p2'  CTRL-P3 = '^p3'

NUM_1 = '\E0q'   NUM_2 = '\E0r'   NUM_3 = '\E0s'   NUM_4 = '\E0t'
NUM_5 = '\E0u'   NUM_6 = '\E0v'   NUM_7 = '\E0w'   NUM_8 = '\E0x'
NUM_9 = '\E0y'   NUM_0 = '\E0p'

}

#-----#
#                                           #
#           vt100 and vt200                #
#           IBM Key <== ASCII key sequence #
#                                           #
#-----#

Enter = Enter | Return          Clear = ESC-c          SysReq = ESC-s
PA-1 = CTRL-P1    PA-2 = CTRL-P2    PA-3 = CTRL-P3
Left=Left | BackSp  Right = Right    Up = Up
Home = ESC-h       Reset = ESC-r      Tab = Tab
Insert = ESC-i     Delete = CTRL-D    Newline = CTRL-N
ErsEof = ESC-e     Attn = CTRL-Z      Dup = ESC-d
FldMark = ESC-f

PF1 = ESC-1        PF2 = ESC-2        PF3 = ESC-3        PF4 = ESC-4
PF5 = ESC-5        PF6 = ESC-6        PF7 = ESC-7        PF8 = ESC-8
PF9 = ESC-9        PF10 = ESC-0       PF11 = ESC--       PF12 = 'ESC= '
PF13 = ESC-!       PF14 = ESC-@       PF15 = 'ESC-#'     PF16 = ESC-$
PF17 = ESC-%       PF18 = ESC-^       PF19 = ESC-&       PF20 = ESC-*
PF21 = ESC-(       PF22 = ESC-)       PF23 = ESC-_       PF24 = ESC-+

Escape = CTRL-X    ClrFld = ESC-SP    Erase = Delete     Help = ESC-?
Status = ESC-o     Reshow = CTRL-L    DmpScrn = ESC-p

cent = ESC-m       not = ESC-n        'brk-|' = ESC-v

0 = 0 | NUM_0      1 = 1 | NUM_1      2 = 2 | NUM_2      3 = 3 | NUM_3
4 = 4 | NUM_4      5 = 5 | NUM_5      6 = 6 | NUM_6      7 = 7 | NUM_7
8 = 8 | NUM_8      9 = 9 | NUM_9

```

Code Example B-1 Sample sun3270map File (7 of 8)

```

=====#
#
#           Keyboard Map Record:           #
#           sun and sun-cmd                #
#                                           #
#                                           #
=====#

sun | sun-cmd
{

#
# Aliases
#
#

aliases
{

    R1 = '\E[208z'   R2 = '\E[209z'   R3 = '\E[210z'   R4 = '\E[211z'
    R5 = '\E[212z'   R6 = '\E[213z'   R7 = '\E[214z'   R8 = '\E[A'
    R9 = '\E[216z'   R10 = '\E[D'      R11 = '\E[218z'   R12 = '\E[C'
    R13 = '\E[220z'  R14 = '\E[B      R15 = '\E[222z'

    L2 = '\E[193z'   F2 = '\E[225z'   F3 = '\E[226z'   F4 = '\E[227z'
    F5 = '\E[228z'   F6 = '\E[229z'   F7 = '\E[230z'   F8 = '\E[231z'
    F9 = '\E[232z'   F10 = '\E[233z'  F11 = '\E[234z'  F12 = '\E[235z'

    Enter= '\E[250z' Delete = '^?'    Del = '\E[249z'   Return = '^m'
    BackSp = '^h'    Tab = '^i'      Ins = '\E[247z'

    ESC-a = 'Ea'     ESC-b = 'Eb'     ESC-c = 'Ec'     ESC-d = 'Ed'
    ESC-e = 'Ee'     ESC-f = 'Ef'     ESC-g = 'Eg'     ESC-h = 'Eh'
    ESC-i = 'Ei'     ESC-j = 'Ej'     ESC-k = 'Ek'     ESC-l = 'El'
    ESC-m = 'Em'     ESC-n = 'En'     ESC-o = 'Eo'     ESC-p = 'Ep'
    ESC-q = 'Eq'     ESC-r = 'Er'     ESC-s = 'Es'     ESC-t = 'Et'
    ESC-u = 'Eu'     ESC-v = 'Ev'     ESC-w = 'Ew'     ESC-x = 'Ex'
    ESC-y = 'Ey'     ESC-z = 'Ez'

    ESC-! = 'E!'     ESC-@ = 'E@'     'ESC-#' = 'E#'   ESC-$ = 'E$'

    ESC-% = 'E%'     ESC-^ = 'E^^'    ESC-& = 'E&'     ESC-* = 'E*'

```

Code Example B-1 Sample sun3270map File (8 of 8)

```

ESC-( = 'E('
ESC-? = 'E?'
CTRL-B = '^b'
CTRL-X = '^x'
CTRL-P1 = '^p1'

ESC-) = 'E)'
CTRL-D = '^d'
CTRL-Z = '^z'
CTRL-P2 = '^p2'

ESC-_ = 'E_'
CTRL-L = '^l'
CTRL-P3 = '^p3'

ESC-+ = 'E+'
CTRL-U = '^u'

#-----#
#
#           sun and sun-cmd
#       IBM Key <== ASCII key sequence
#
#-----#

Enter = Enter
PA-1 = CTRL-P1
Left= R10 | BackSp
Home = R7
Insert = Ins
ErsEof = R1

Clear = ESC-c
PA-2 = CTRL-P2
Right = R12
Reset = R11
Delete = Del
Attn = R3

SysReq = ESC-s
PA-3 = CTRL-P3
Up = R8
Tab = Tab
Newline = Return
Dup = ESC-d

Down = R14
Btab = CTRL-B
ErsInpt = CTRL-U
FldMark = ESC-f

PF1 = L2
PF5 = F5
PF9 = F9
PF13 = ESC-!
PF17 = ESC-%
PF21 = ESC-(

PF2 = F2
PF6 = F6
PF10 = F10
PF14 = ESC-@
PF18 = ESC-^
PF22 = ESC-)

PF3 = F3
PF7 = F7
PF11 = F11
PF15 = 'ESC-#'
PF19 = ESC-&
PF23 = ESC-_

PF4 = F4
PF8 = F8
PF12 = F12
PF16 = ESC-$
PF20 = ESC-*
PF24 = ESC-+

Escape = CTRL-X
Help = ESC-?

cent = ESC-m

ClrFld = ESC-e
Status = ESC-o

not = ESC-n

Erase = R13 | Delete
Reshow = CTRL-L
'brk-|' = ESC-v

DmpScrn = R2

```

Using PCFT



SUNWopc1 supports the transfer of binary, ASCII, and EBCDIC text files between the local Unix system and IBM host systems that support the IBM 3270 PC file transfer option. You can transfer files to and from IBM host systems that support:

- CICS (Customer Information Control System)—CICS/VS 5798-DQH, Version 1.00 or greater
- TSO (Time Sharing Option)—MVS 5665-311, Version 1.00 or greater
- CMS (Conversational Monitoring System)—VM/SP 5664-281, Version 1.00 or greater

pcft starts the Sun IBM 3270 PC file transfer (IND\$FILE) process.

Note – Pacific Rim users: SUNWopc1 also supports the transfer of DBCS text files. If `dbcsMode` is turned on, the default for the File Transfer process is `APVUFILE` rather than `(IND$FILE)`

Two options indicate the direction of the transfer:

- `send` initiates a transfer from the local system to the IBM host system.
- `receive` initiates a transfer from the IBM host system to the local system.

Before you initiate a file transfer, you must be logged into one of the above-mentioned IBM host applications and positioned at the “main” prompt for that IBM host application. In TSO, for example, you should have the “READY” prompt displayed in your SUNWopc1 window.

To execute a file transfer:

- 1. Establish a communications link with the IBM host system.**
- 2. Start SUNWopc1 as an EHLLAPI server: use the `-e session_name` to identify the EHLLAPI session name.**
- 3. Log into the IBM host application (TSO, CICS, or VM/CMS).**
- 4. Ensure that the “main” prompt for the IBM host application is displayed.**
- 5. Move to a Unix shell window.**
- 6. For sun3270tty, use the Escape-to-shell key sequence to invoke a shell on your TTY display.**
After executing the `pcft` command, you can return to your 3270 emulation window by exiting from the shell.
- 7. Issue the `pcft` (send or receive) command.**

`pcft` can also be used for exchanging several files with the IBM host. Use `pcft` in a shell script. Several `pcft` commands can be executed sequentially by the shell script. This batch file transfer is very useful for moving files between your local system and the IBM host (see Table C-1).

♦ **To invoke `pcft`, type the following command line.**

```
pcft send|receive local_file <n:>host_file <options>
```

The different options are described separately for each IBM host application in subsequent sub-sections).

Table C-1 `pcft` Command Options

Option	Function
<code>send</code>	Sends file to IBM host
<code>receive</code>	Receives file from IBM host
<code>local_file</code>	Path name of file on local system
<code>n</code>	The LU session to use for transfer (EHLLAPI session name (A-H))

Table C-1 pcft Command Options (Continued)

Option	Function
<i>host_file</i>	File name on IBM host
<i>options</i>	Optional values controlling file transfer, each application has its own option format (see below)
< >	Indicates optional parameter

Note – To prevent the Unix shell from interpreting the *options* parameter for this program, include all the *options* in double quotes “ ”. The shell will not attempt to interpret ((single left parenthesis). *pcft* will not include the double quotes in your request to the IBM host application. Pacific Rim users, see Appendix F, “DBCS,” for additional keywords that apply to *pcft*.

C.1 TSO Options

This section describes the options that are available for file transfers between the Sun *pcft* program and a TSO host application. Remember to move to the TSO READY prompt before initiating a file transfer.

Note – If the IBM host system is running TSO-E, you must disable interrupts and exit from the session manager before beginning a file transfer. You can disable TSO interrupts with the *term nobreak* command. You can exit from the TSO session manager by clearing the screen and entering *end*.

pcft cannot create a partitioned data set under TSO. *pcft*, however, can create new members in an existing partitioned data set.

TSO automatically prefixes the TSO data set name and member name with your TSO User ID. Use single quotation marks (') to avoid having the User ID prefixed to the TSO data set name and member name you specify.

C.1.1 Send (TSO)

The following command describes the Send (TSO) command. The syntax of the *pcft* command that transfers files from the SUNWopc1 session to a TSO host application is described in Table C-2:

```
pcft send local_file <n:>host_file<(member_name)></password> <options>
```

Options: append ascii blksize(*n*) crlf lrecl(*n*) recfm(*f|v|u*)
space(*q<,i>*) avblock(*x*) tracks cylinders

Table C-2 Send (TSO) Syntax

Element	Description
send	Send file to IBM host application. Required positional parameter.
<i>local_file</i>	Pathname of file on local system. Required positional parameter.
<i>n</i>	<i>n</i> is the LU session to use for transfer (EHLLAPI session name (A-H)). Optional positional parameter. Default is A.
<i>host_file</i>	TSO data set name on IBM host. Refer to the <i>IBM OS/VS2 Command Language Reference</i> manual for more information about TSO data set names and properties. Required positional parameter.
(<i>member_name</i>)	Name of member in the TSO data set specified by <i>host_file</i> . This is an optional positional parameter. Follows <i>host_file</i> immediately, with no space separators.
/password	Password used for TSO data set under password-protection. This is an optional positional parameter. Follows <i>host_file</i> (<i>member_name</i>) immediately, with no space separators.
options	TSO supports the following send options:
append	Appends the contents of the <i>local_file</i> to the TSO data set. append cannot be used for members of a partitioned data set. The logical record length and record format values are the same as the original TSO data set. Optional parameter.
ascii	Indicates to TSO that the <i>local_file</i> should be converted from ASCII to EBCDIC when stored in the TSO data set. Use this option, along with crlf , for readable files. Do not use this option for binary files. Default: TSO does not translate the file from ASCII to EBCDIC.
blksize(<i>n</i>)	Indicates desired blocks size for the host data set (in bytes). Default: 80 bytes for new files; existing blocksize for existing files.
crlf	Indicates that carriage return/line feed characters (CR/LF) act as record separators in the data transferred to the TSO application. TSO removes the CR/LF characters and marks the end of each record with its own internal end-of-record marker. With this option, pcft reads each line of text from the <i>local_file</i> and marks the end of each line with CR/LF characters before transmitting the data to the TSO application. Use this option, along with ASCII, for readable files. Do not use this option for binary files. Default: TSO does not insert end of record markers for CR/LF characters.
lrecl(<i>n</i>)	Indicates the desired logical record length for the TSO data set (in bytes). Default: 80 bytes for new files; existing logical record length for existing files.
recfm(<i>f v u</i>)	Indicates the desired record format for the TSO data set, where:
f	Specifies fixed-length records.
v	Specifies variable-length records.

Table C-2 Send (TSO) Syntax (Continued)

Element	Description
u	Specifies undefined-length records. Default: Fixed-length records for new files (without <code>cr lf</code> option specified); variable-length records for new files (with <code>cr lf</code> option specified).
space(<i>q</i> ,< <i>i</i> >)	Indicates the space allocation requirements for a new TSO data set, where <i>q</i> is the primary and <i>i</i> is secondary allocation of <i>avblock</i> , <i>tracks</i> , and <i>cylinders</i> , and where
< <i>avblock</i> (<i>x</i>)>	Is the smallest entity
< <i>tracks</i> >	Is middle-sized entity
< <i>cylinders</i> >	Is the largest entity.
	Default: default allocation value of <code>blksize</code> .

C.1.2 Send (TSO) Examples

To send a text file (name: `textfile`) from the local Unix system to a host TSO data set (data set name: `data.abc`) in readable format, type:

```
% pcft send textfile data.abc ascii cr lf
```

To send a binary file (name: `binfile`) from the local Unix system to a host TSO data set (data set name: `data.def`, member name: `binfile`), type:

```
% pcft send binfile data.def(binfile) recfm(v)
```

To send a binary file (name: `binfile`) from the local Unix system to a host TSO data set (data set name: `sys.tables`, member name: `mystuff`) and indicate that your TSO User ID should not be prefixed to the indicated data set name, type:

```
% pcft send binfile "'sys.tables(mystuff)'" recfm(v)
```

or,

```
% pcft send binfile \'sys.tables\'(mystuff)\' recfm(v)
```

The double quotes (“) protect the data set and member name from being interpreted by the Unix shell. Similarly, the back slashes (\) notify the Unix shell not to interpret the next characters.

C.1.3 Receive (TSO)

The syntax of the `pcft` command to transfer files from a TSO host application to a `SUNWopc1` session, which is shown below, is described in Table C-3.

```
pcft receive local_file <n:>host_file<(member_name)></password> <options>
```

Options: `append` `ascii` `crlf`

Table C-3 Receive (TSO) Syntax

Element	Description
<code>receive</code>	Receive file from IBM host application. Required positional parameter.
<code>local_file</code>	Pathname of file on local system. Required positional parameter.
<code>n</code>	<code>n</code> is LU session to use for transfer (EHLLAPI session name (A-H)). Optional positional parameter. Default is 'A'.
<code>host_file</code>	TSO data set name on IBM host. Refer to the <i>IBM OS/VS2 Command Language Reference</i> manual for more information about TSO data set names and properties. Required positional parameter.
<code>(member_name)</code>	Name of member in the TSO data set specified by <code>host_file</code> . Optional positional parameter. Follows <code>host_file</code> immediately, no space separators.
<code>/password</code>	Password used for TSO data set under password-protection. Optional positional parameter. Follows <code>host_file(member_name)</code> immediately, no space separators.
<code>options</code>	TSO supports the following receive options:
<code>append</code>	Appends the contents of the TSO data set to the <code>local_file</code> . Optional parameter.
<code>ascii</code>	Indicates to TSO that the TSO dataset should be converted from EBCDIC to ASCII when transferred to the <code>local_file</code> . Use this option, along with <code>crlf</code> , for readable files. Do not use this option for binary files. Default: TSO does not translate the file from EBCDIC to ASCII.
<code>crlf</code>	Indicates that carriage return /line feed characters (CR/LF) act as record separators in the data transferred from the TSO application to the <code>local_file</code> . TSO adds the CR/LF characters where its own internal end-of-record markers indicated end of records. With this option, <code>pcft</code> replaces the CR/LF characters with a LF character before writing each text line to the <code>local_file</code> . Use this option, along with ASCII, for readable files. Do not use this option for binary files. Default: TSO does not insert CRLF characters to indicate end of record.

C.1.4 Receive (TSO) Examples

To receive a host TSO data set (data set name: `data.abc`) as a text file (name: `textfile`) in readable format on your Unix system, type:

```
% pcft receive textfile data.abc ascii crlf
```

To receive a host TSO data set (data set name: `data.def`, member name: `binfile`) as binary file (name: `binfile`) on your Unix system, type:

```
% pcft receive binfile data.def(binfile)
```

To receive a host TSO data set (data set name: `data.def`, member name: `binfile`) as a binary file (name: `binfile`) on your Unix system and indicate that your TSO User ID should not be prefixed to the indicated data set name, type:

```
% pcft receive binfile "'sys.tables(mystuff)'"
```

or,

```
% pcft receive \'sys.tables\(\mystuff)\\'
```

The double quotes (“”) protect the data set and member name from being interpreted by the Unix shell. Back slashes (\) notify the Unix shell that the characters that follow, should not be interpreted.

C.2 CICS Options

This section describes the options available for file transfers between Sun's `pcft` program and a CICS host application. Remember to move the CICS READY prompt before initiating a file transfer.

Note – The `pcft` cannot create a partitioned data set under CICS. However, `pcft` can create new files when the data set is not partitioned.

C.2.1 Send (CICS)

The syntax of the `pcft` command to transfer files from the `SUNWopc1` session to a CICS host application, which is shown below, is described in Table C-4.

```
pcft send local_file <n:>host_file<(options)< >comments>
```

Options: `ascii` `crlf`

Table C-4 Send (CICS) Syntax

Element	Description
<code>send</code>	Send file to IBM host application. Required positional parameter.
<code>local_file</code>	Pathname of file on local system. Required positional parameter. Pacific Rim Users: If <code>dbcsMode</code> is set, <code>pcft</code> translates the data set from DBCS to EUC.
<code>n</code>	<code>n</code> is the LU session to use for transfer (EHLLAPI session name (A-H)). Optional positional parameter. Default is 'A'.
<code>host_file</code>	CICS data set name on IBM host. The data set name can be up to 8 characters long. Refer to the <i>IBM Customer Information Control System (CICS/OS/VS) Version 1 Release 6 Installation and Operations Guide</i> manual for more information about CICS data set names and properties. Required positional parameter.
<code>(options</code>	The (character is mandatory when you specify options. CICS supports the following send options:
<code>ascii</code>	Indicates to CICS that the <code>local_file</code> should be converted from ASCII to EBCDIC when stored in the CICS data set. Use this option, along with <code>crlf</code> , for readable files. Do not use this option for binary files. Default: CICS translates the file from ASCII to EBCDIC.
<code>binary</code>	Indicates that CICS should not attempt to convert the <code>local_file</code> data from ASCII to EBCDIC. Default: CICS translates the file from ASCII to EBCDIC. NOTE: Pacific Rim Users, if <code>dbcsMode</code> is set, <code>pcft</code> translates the data set from DBCS to EUC.
<code>crlf</code>	Indicates that carriage return /linefeed characters (CR/LF) act as record separators in the data transferred to the CICS application. CICS removes the CR/LF characters and marks the end of each record with its own internal end-of-record marker. With this option, <code>pcft</code> reads each line of text from the <code>local_file</code> and marks the end of each line with CR/LF characters before transmitting the data to the CICS application. Use this option, along with <code>ASCII</code> , for readable files. Do not use this option for binary files. Default: CICS inserts end of record markers for CR/LF characters.
<code>no_crlf</code>	Indicates that carriage return /line feed characters (CR/LF) do not act as record separators. Default: CICS inserts end of record markers for CR/LF characters.
<code>)comments</code>	Lets you add information regarding the file transfer. The (character is mandatory; when you do not specify options, indicate their absence by entering () before your comments.

C.2.2 Send (CICS) Examples:

- ♦ **To send a text file (name: textfile) from the local Unix system to a host CICS data set (data set name: data) in readable format, type:**

```
% pcft send textfile data (ascii crlf
```

- ♦ **To send a binary file (name: binfile) from the local Unix system to a host CICS data set (data set name: data1), type:**

```
% pcft send binfile data1 (binary nocrlf
```

C.2.3 Receive (CICS)

The syntax of the pcft command to transfer files from a CICS host application to a SUNWopc1 session is described in Table C-5:

```
pcft receive local_file <n:>host_file <(options)>< )comments>
```

Options: ascii crlf

Table C-5 Receive (CICS) Syntax

Element	Description
receive	Receive file from IBM host application. Required positional parameter.
local_file	Pathname of file on local system. Required positional parameter.
n	n is LU session to use for transfer (EHLLAPI session name (A-H)). Optional positional parameter. Default is A.
host_file	CICS data set name on IBM host. The data set name can be up to 8 characters. Refer to the IBM <i>Customer Information Control System (CICS/OS/VS) Version 1 Release 6 Installation and Operations Guide</i> manual for more information about CICS data set names and properties. Required positional parameter.
(options	The (character is mandatory when you specify options. CICS supports the following receive options: ascii Indicates to CICS that the dataset should be converted from EBCDIC to ASCII when transferred to the local_file. Use this option, along with crlf, for readable files. Do not use this option for binary files. Default: CICS translates the file from EBCDIC to ASCII.

Table C-5 Receive (CICS) Syntax (Continued)

Element	Description
binary	Indicates that CICS should not attempt to convert the data set from EBCDIC to ASCII. Default: CICS translates the data set from EBCDIC to ASCII. Pacific Rim Users: If dbcsMode is set, pcft translates the data set from DBCS to EUC.
crlf	Indicates that carriage return/line feed characters (CR/LF) act as record separators in the data transferred from the CICS application to the <i>local_file</i> . CICS adds the CR/LF characters where its own internal end-of-record markers indicated end of records. With this option, pcft replaces the CR/LF characters with a LF character before writing each text line to the local file. Use this option, along with ASCII, for readable files. Do not use this option for binary files. Default: CICS inserts CR/LF characters to indicate end of record.
no_crlf	Indicates that carriage return/line feed characters (CR/LF) do not act as record separators. Default: CICS inserts end of record markers for CR/LF characters.
)comments	Lets you add information regarding the file transfer. The) character is mandatory; when you do not specify options, indicate their absence by entering () before your comments.

C.2.4 Receive (CICS) Examples

- ♦ **To receive a host CICS data set (data set name: data) as a text file (name: textfile) in readable format on your Unix system, type:**

```
% pcft receive textfile data (ascii crlf
```

- ♦ **To receive a host CICS data set (data set name: data1) as binary file (name: binfile) on your Unix system, type:**

```
% pcft receive binfile data1 (binary nocrlf
```

C.3 VM/CMS Options

This section describes the options available for file transfers between Sun's pcft program and a VM/CMS host application. Remember to move to the VM/CMS READY prompt before initiating a file transfer.

C.3.1 Send (VM/CMS)

The syntax of the `pcft` command to transfer files from the `SUNWopc1` session to a VM/CMS host application is described in Table C-6.

```
pcft send local_file <n:>host_file host_filetype <hostfilemode _<(options>
```

Options: `append ascii crlf lrecl n recfm f|v`

Table C-6 Send (VM/CMS) Syntax

Element	Description
<code>send</code>	Send file to IBM host application. Required positional parameter
<code>local_file</code>	Pathname of file on local system. Required positional parameter
<code>n</code>	<code>n</code> is LU session to use for transfer (EHLLAPI session name (A-H)). Optional positional parameter. Default is A.
<code>host_file</code>	VM/CMS file name on IBM host. File name can be up to 8 characters. Refer to the <i>IBM Virtual Machine/System Product: CMS User's Guide</i> for more information about VM/CMS file names and properties. Required positional parameter
<code>file_type</code>	VM/CMS file type. Required positional parameter
<code>file_mode</code>	VM/CMS file mode. Optional positional parameter Default is A1.
<code>(options</code>	The (character is mandatory when you specify options. VM/CMS supports the following send options:
<code>append</code>	Appends the contents of the <code>local_file</code> to the VM/CMS file. The logical record length and record format values are the same as the original VM/CMS file. Optional parameter.
<code>ascii</code>	Indicates to VM/CMS that the <code>local_file</code> should be converted from ASCII to EBCDIC when stored in the VM/CMS file. Use this option, along with <code>crlf</code> , for readable files. Do not use this option for binary files. Default: VM/CMS does not translate the file from ASCII to EBCDIC.
<code>crlf</code>	Indicates that carriage return/line feed characters (CR/LF) act as record separators in the data transferred to the VM/CMS application. VM/CMS removes the CR/LF characters and marks the end of each record with its own internal end-of-record marker. With this option, <code>pcft</code> reads each line of text from the local file and marks the end of each line with CR/LF characters before transmitting the data to the TSO application. Use this option, along with <code>ascii</code> , for readable files. Do not use this option for binary files. Default: VM/CMS does not insert end of record markers for CR/LF characters.
<code>lrecl n</code>	Indicates the desired logical record length for the VM/CMS file (in bytes). Default: 80 bytes for new files; existing logical record length for existing files.

Table C-6 Send (VM/CMS) Syntax (Continued)

Element	Description
lrecl <i>n</i>	Indicates the desired logical record length for the VM/CMS file (in bytes). Default: 80 bytes for new files; existing logical record length for existing files.
recfm <i>f v</i>	Indicates the desired record format for the VM/CMS file, where <i>f</i> Specifies fixed-length records <i>v</i> Specifies variable-length records Default: Fixed-length records for new files (without <code>cr lf</code> option specified); variable-length records for new files (with <code>cr lf</code> option specified).

C.3.2 Send (VM/CMS) Examples

- ♦ **To send a text file (name: `textfile`) from the local Unix system to a host VM/CMS file (name: `data`) in readable format, type:**

```
% pcft send textfile data basic a1 (ascii crlf
```

- ♦ **To send a binary file (name: `binfile`) from the local Unix system to a host VM/CMS file (name: `data1`), type:**

```
% pcft send binfile data1 basic a1 (recfm v
```

C.4 Receive (VM/CMS)

The syntax of the `pcft` command to transfer files from a VM/CMS host application to a `SUNWopc1` session is described in Table C-7 below:

```
pcft receive local_file <n:>host_file host_filetype <host_filemode> _<(options>
```

Options: `append` `ascii` `crlf`

Table C-7 Receive (VM/CMS) Syntax

Element	Description
<code>receive</code>	Receive file from IBM host application. Required positional parameter.
<code>local_file</code>	Pathname of file on local system. Required positional parameter.
<code>n</code>	<code>n</code> is LU session to use for transfer (EHLLAPI session name (A-H)). Optional positional parameter. Default is A.
<code>host_file</code>	VM/CMS file name on IBM host. File name can be up to 8 characters. Refer to the <i>IBM Virtual Machine/System Product: CMS User's Guide</i> manual for more information about VM/CMS file names and properties. Required positional parameter.
<code>file_type</code>	VM/CMS file type. Required positional parameter.
<code>file_mode</code>	VM/CMS file mode. Optional positional parameter. Default is A1.
<code>(options</code>	The (character is mandatory when you specify options. VM/CMS supports the following send options:
<code>append</code>	Appends the contents of the VM/CMS file to the <code>local_file</code> . Optional parameter.
<code>ascii</code>	Indicates to VM/CMS that the VM/CMS file should be converted from EBCDIC to ASCII when transferred to the local file. Use this option, along with <code>cr lf</code> , for readable files. Do not use this option binary files. Default: VM/CMS does not translate the file from EBCDIC to ASCII.
<code>cr lf</code>	Indicates that carriage return/linefeed characters (CR/LF) act as record separators in the data transferred from the VM/CMS application to the local file. VM/CMS adds the CR/LF characters where its own internal end-of-record markers indicated end of records. With this option, <code>pcft</code> replaces the CR/LF characters with a LF character before writing each text line to the local file. Use this option, along with ASCII for readable files. Do not use this option for binary files. Default: VM/CMS does not insert CRLF characters to indicate end of record.

C.4.1 Receive (VM/CMS) Examples

- ♦ **To receive a host VM/CMS file (name: data) as a text file (name: textfile) in readable format on your Unix system, type:**

```
% pcft receive textfile data basic a1 (ascii cr lf
```

- ♦ **To receive a host VM/CMS file (name: data1) as binary file (name: binfile) on your Unix system, type:**

```
% pcft receive binfile data1 basic a1
```


SunLink3270 Tracing



To aid you in troubleshooting, sun3270 has internal tracing capabilities. To turn on the trace utility, start the appropriate emulator with the `-t` option. For example to start sun3270x with tracing enabled, use the following command:

```
# sun3270x -t -1
```

The sun3270tty, suntn3270x and suntn3270tty emulators can all be started in this way. The `t` option with the `-1` argument requests sun3270 to dump all encountered internal trace points. Chapter 3, “Using sun3270x” describes sun3270 start options. sun3270 records all trace points in the `sunlib_pid` file in the current working directory. This file is reused after recording 1000 trace points (the older trace information is saved in `sunlib_pid.1`).

The trace points record all information received and sent to the sunpu2.1 SNA server, and all information received and sent to the terminal.

Trace points have this format:

```
07/26/96 10:50:21 fsm automation:
Screen owner = GR_SSCP_LU State = S0 Event = EV_OPEN_LU
```

Each trace point contains the following information:

- Date and time—time trace point recorded (in seconds)
- Component—the device recording this trace point
- Information string—general description of type of trace point
- Hexadecimal message dump or information string (optional)

The most common types of trace points are:

- `rcv_ru`: Receives message from SNA host through `sunpu2.1` SNA server
- `send_ru`: Sends message to SNA host through the `sunpu2.1` SNA server
- `fsm_automation`: Process events captured by `sun3270`
- `read_from_terminal`: Inputs data read from the terminal

An example of trace points is listed and described in Code Example D-1.

Code Example D-1 sun3270 Trace (1 of 9)

```
07/26/96 10:50:21 start_screen: LI
LI = 25

07/26/96 10:50:21 start_screen: CO
CO = 81

        # Sun3270 requests connection to sunpu2.1
        # SNA Server
07/26/96 10:50:21 sun_open_port: open request
0002000000000000 0000000000000000 0000000000000000 .....
0000000000000000 0000000000000000 0000000000000000 .....
0000000000000000 0000000000000000 0000000000000000 .....

        # Sun3270 receives open connection response
        # from sunpu2.1 SNA Server
07/26/96 10:50:21 open_lu_rsp: sun_open_type
0000000100000000 0000000700000000 0000000000000000 .....
0000000000000000 534c553331000000 5350363300000000 SLU31...SP63....
00020000 .....

        # Sun3270 begins processing event from server
07/26/96 10:50:21 fsm automation:
Screen owner = GR_SSCP_LU State = S0 Event = EV_SERV_IN

        # Sun3270 recognizes event as open connection
07/26/96 10:50:21 fsm automation:
Screen owner = GR_SSCP_LU State = S0 Event = EV_OPEN_LU

        # Sun3270 sees that the SSCP-LU session is
        # active
07/26/96 10:50:21 fsm automation:
Screen owner = GR_SSCP_LU State = S0 Event = EV_SSCP_LU_ACT
```

Code Example D-1 sun3270 Trace (2 of 9)

```
# Sun3270 receives a logon banner from the
# SNA Host. Next two messages are for
# the banner message: the first shows the
# control and RH indicators and the second
# shows the 3270 data stream. Only part of
# the banner message is shown.
07/26/96 10:50:31 rcv_ru: sun_std_type
000000004000000000 000000007000000000 818000000000000000
00000000000020000

07/26/96 10:50:31 rcv_ru: ru
e6c5d3c3d6d4c540 e3d640e3c8c540c9 c2d440e3c5e2e340      WELCOME TO THE IBM TEST
# ...
40c1d7d7d3c9c44d e75d40d3d6c7d4d6 c4c54de85d40c4c1      APPLID(X)/LOGMODE(Y) DA
e3c14de95d157e7e 7e6e40                                     TA(Z).==>

# Sun3270 begins processing event from server
07/26/96 10:50:31 fsm automation:
Screen owner = GR_SSCP_LU  State = S1  Event = EV_SERV_IN

# Sun3270 recognizes banner as FMD data
07/26/96 10:50:31 fsm automation:
Screen owner = GR_SSCP_LU  State = S1  Event = EV_FMD

# Sun3270 sends positive response to banner
07/26/96 10:50:31 sun_accept_ru: sun_std_type

# Sun3270 sees end of chain on message
07/26/96 10:50:31 fsm automation:
Screen owner = GR_SSCP_LU  State = S4  Event = EV_EC

# Sun3270 receives input from terminal keyboard
07/26/96 10:50:33 read_from_terminal: keyboard
6e                                     n

# Sun3270 recognizes input as terminal input
07/26/96 10:50:33 fsm automation:
Screen owner = GR_SSCP_LU  State = S1  Event = EV_TERM_IN

# Sun3270 processes this input
07/26/96 10:50:33 fsm automation:
Screen owner = GR_SSCP_LU  State = S1  Event = EV_FIRST_KEY
```

Code Example D-1 sun3270 Trace (3 of 9)

```
# Sun3270 receives input from terminal keyboard
07/26/96 10:50:34 read_from_terminal: keyboard
65                                     e

# Sun3270 recognizes input as terminal input
07/26/96 10:50:34 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

# Sun3270 processes this input
07/26/96 10:50:34 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_FIRST_KEY

07/26/96 10:50:34 read_from_terminal: keyboard
74                                     t

07/26/96 10:50:34 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:34 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_FIRST_KEY

07/26/96 10:50:35 read_from_terminal: keyboard
76                                     v

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_FIRST_KEY

07/26/96 10:50:35 read_from_terminal: keyboard
69                                     i

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_FIRST_KEY

07/26/96 10:50:35 read_from_terminal: keyboard
65                                     e
```

Code Example D-1 sun3270 Trace (4 of 9)

```
07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_FIRST_KEY

07/26/96 10:50:35 read_from_terminal: keyboard
77                                     w

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:35 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_FIRST_KEY

        # Sun3270 receives input from terminal keyboard
07/26/96 10:50:38 read_from_terminal: keyboard
1b                                     .

        # Sun3270 recognizes input as terminal input
07/26/96 10:50:38 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

        # Sun3270 receives input from terminal keyboard
07/26/96 10:50:38 read_from_terminal: keyboard
5b                                     [

        # Sun3270 recognizes continuing input from
        # terminal input ...
07/26/96 10:50:38 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:38 read_from_terminal: keyboard
32                                     2

07/26/96 10:50:38 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:38 read_from_terminal: keyboard
35                                     5

07/26/96 10:50:38 fsm automation:
```

Code Example D-1 sun3270 Trace (5 of 9)

```
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:38 read_from_terminal: keyboard
30                                         0

07/26/96 10:50:38 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

07/26/96 10:50:38 read_from_terminal: keyboard
7a                                         z

07/26/96 10:50:38 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_TERM_IN

        # Sun3270 recognizes input data from terminal
        # as the mapped ENTER key
07/26/96 10:50:38 fsm automation:
Screen owner = GR_SSCP_LU  State = S2  Event = EV_ENTER_KEY

        # Sun3270 sends input data to SNA Host.
        # Data traced as two messages: the first
        # message represents control and RH
        # indicators, and the second is the 3270
        # data stream. Logon to NetView.
07/26/96 10:50:38 sun_send_ru: sun_std_type
0000000000000000 0000000700000000 8180000000000000
0000000010000000

07/26/96 10:50:38 sun_send_ru: ru
9585a3a58985a6                               netview

        # Sun3270 receives reply from sunpu2.1 SNA
        # Server stating message received OK
07/26/96 10:50:39 reply: sun_std_type
0000000030000000 0000000700000000 0000000000000000
0000000000004000

        # Sun3270 recognizes the server reply and
        # processes it
07/26/96 10:50:39 fsm automation:
Screen owner = GR_SSCP_LU  State = S3  Event = EV_SERV_IN
```

Code Example D-1 sun3270 Trace (6 of 9)

```
# Sun3270 receives a positive SNA response
# from the SNA Host for logon message
07/26/96 10:50:39 rcv_ru: sun_std_type
0000000500000000 0000000700000000 8180000000000000
0000000000040000

# Sun3270 recognizes event from server
07/26/96 10:50:39 fsm automation:
Screen owner = GR_SSCP_LU State = S3 Event = EV_SERV_IN

# Sun3270 process the positive SNA response
07/26/96 10:50:39 fsm automation:
Screen owner = GR_SSCP_LU State = S3 Event = EV_POS_RSP

# Sun3270 receives logon message. Again,
# two trace points: control and data.
07/26/96 10:50:39 rcv_ru: sun_std_type
0000000400000000 0000000700000000 8180000000000000
0000000000030000

07/26/96 10:50:39 rcv_ru: ru
15d3d6c7d6d540d9 c5d8e4c5e2e340d8 e4c5e4c5c440e3d6 .LOGON REQUEST QUEUED TO
40c1d7d7d3c9c3c1 e3c9d6d515 APPLICATION.

07/26/96 10:50:39 fsm automation:
Screen owner = GR_SSCP_LU State = S1 Event = EV_SERV_IN

07/26/96 10:50:39 fsm automation:
Screen owner = GR_SSCP_LU State = S1 Event = EV_FMD

# Sun3270 sends positive response to logon
# message
07/26/96 10:50:39 sun_accept_ru: sun_std_type

07/26/96 10:50:39 fsm automation:
Screen owner = GR_SSCP_LU State = S4 Event = EV_EC

# Sun3270 receives a BIND (0x31) request from
# SNA Host. The first trace point shows the
# control, RH indicators, and the RU code
# (0x31). The second trace point shows the
# RU data -- everything after the RU code in
```

Code Example D-1 sun3270 Trace (7 of 9)

```

# an SNA message.
07/26/96 10:50:52 rcv_ru: sun_std_type
000000004000000000 000000007000000031 4180800000010000
00000000003490000

07/26/96 10:50:52 rcv_ru: ru
010303b190308000 0187c70100020000 000000185000007e .....gG.....&..=
000005c3d5d4f0f1 00 ...CNM01.

07/26/96 10:50:52 fsm automation:
Screen owner = GR_SSCP_LU State = S1 Event = EV_SERV_IN

# Sun3270 recognizes the BIND request
07/26/96 10:50:52 fsm automation:
Screen owner = GR_SSCP_LU State = S1 Event = EV_BIND

# Sun3270 changes the Screen owner
07/26/96 10:50:52 fsm automation:
Screen owner = GR_LU_LU State = S0 Event = EV_BIND

# Sun3270 sends positive SNA response to BIND
07/26/96 10:50:52 sun_accept_ru: sun_std_type

# Some trace points not shown ...

# Sun3270 receives the first chain of the
# main NetView logon screen. Again,
# two trace points: control and data.
07/26/96 10:50:57 rcv_ru: sun_std_type
000000004000000000 000000007000000000 8180300000010000
0000000000010000

07/26/96 10:50:57 rcv_ru: ru
f54011c5c4d5d540 404040d5d511c6d4 d5d5d5404040d5d5 5 .EDNN NN.FMNNN NN
11c7e4d5d5d5d540 40d5d511c8f4d5d5 40d5d540d5d5114a GUNNNN NN.H4NN NN NN.
c4d5d54040d5d5d5 d5114bd4d5d54040 40d5d5d5114ce4d5 DNN NNNN..MNN NNN.<UN
d540404040d5d511 c65fc5c5c5c5c5c5 11c76fc5c511c87f N NN.F^EEEEEE.G?EE.H"
c5c5c5c5114a4fc5 c5114b5fc5c5c5c5 c5c511c6e7e3e3e3 EEEE.^EEEEEE.FXTTT
e3e3e3e3e311c77a e3e311c94ae3e311 4a5ae3e3114b6ae3 TTTT.G:TT.IT
e311c560e5e54040 4040404040404040e5 e511c6f1e5e54040 T.E-VV VV.F1VV
4040404040e5e511 c8c2e5e540404040 40e5e511c9d3e5e5 VV.HBVV VV.ILVV
404040e5e5114ae4 e5e540e5e5114bf5 e5e5e5114dc6e511 VV.VV..5VVV.(FV.

```

Code Example D-1 sun3270 Trace (8 of 9)

```
c67fc9c911c84fc9 c911c95fc9c9114a 6fc9c9114b7fc9c9 F"II.H|II.I^II."II
11c7c4c5c5c5c5c5 c511c8d4c5c511c9 e4c5c5c5c5114af4 .GDEEEEEEE.HMEE.IUEEEE.
c5c5114cc4c5c5c5 c5c5c511c74ce6e6 40404040404040 EE.<DEEEEEEE.G<WW
40e6e64040e3d411 c85ce6e640404040 e640404040e6e611 WW TM.H*WW W WW.
c96de6e64040e6e6 e64040e6e6114a7e e6e6e6e640e6e6e6 I_WW WWW WW.WWW
e6114c4fe6e64040 40e6e6114f4e4dc3 5d40c39697a89989 W.<|WW WW.|+(C) Copyright IBM.
8788a340c9c2d411 4f60c39699974b40 f1f9f8f7406040c1 |-Corp. 1987 - All.|1Rights Reserved.
9393114ff1d98987 88a3a240d985a285 99a585841150e5d3 &VLicensed Material.&8-
89838595a2858440 d481a38599898193 1150f86040d79996 Property of IBM.A(
978599a3a8409686 40c9c2d411c14d40 4040
```

07/26/96 10:50:57 fsm automation:
Screen owner = GR_LU_LU State = S2 Event = EV_SERV_IN

Sun3270 recognizes Begin Bracket (BB)
indication in this first message

07/26/96 10:50:57 fsm automation:
Screen owner = GR_LU_LU State = S2 Event = EV_BB

Sun3270 sends a positive SNA response for
the first message

07/26/96 10:50:57 sun_accept_ru: sun_std_type

Sun3270 recognizes End Bracket (EB)
indication in this first message

07/26/96 10:50:57 fsm automation:
Screen owner = GR_LU_LU State = S4 Event = EV_EB

Sun3270 receives notification from sunpu2.1
SNA Server that LU-LU session is between
brackets

07/26/96 10:50:58 notification: sun_std_type
000000007000000000 00000000701000001 0000000000000000
0000000000000000

Some trace points not shown ...

Sun3270 receives an UNBIND (0x32) request
from the application. Again message
with two trace points: control and data.

Code Example D-1 sun3270 Trace (9 of 9)

```
07/26/96 10:51:02 rcv_ru: sun_std_type
0000000700000000 0000000700000032 4180800000010000
00000000034e0000

07/26/96 10:51:02 rcv_ru: ru
01

07/26/96 10:51:02 fsm automation:
Screen owner = GR_LU_LU  State = S2  Event = EV_SERV_IN

      # Sun3270 recognizes event as an UNBIND
07/26/96 10:51:02 fsm automation:
Screen owner = GR_LU_LU  State = S2  Event = EV_UNBIND
```

Error Messages



All SunLink 9.1 IBM connectivity products are distributed with a message database called **BMD**. All informational and error messages output by Sun software are derived from the **BMD**. Product components running in “user space” access the **BMD** during runtime. Product components running in “kernel space” access message tables that are constructed at compilation time.

The Sun message handling scheme provides the following advantages:

- All messages output by Sun software are documented. When a message is output, users have a reference that describes the reason the message was output and an explanation of its impact and how to respond.
- Online message documentation. **BMD** messages are standard Unix text files and may be browsed using normal system editors and utilities, such as **grep**. Sun also distributes an application, **bmsg**, that, given a message number, prints the full content of the message.
- Configurable message formats are used. Users are free to modify the message content as they wish.
- Consistent message handling for all SunLink IBM connectivity products

The use of **bmsg** is described in the following section which also describes the structure of the **BMD** and the format of the message entries.

E.1 bmsg: Sun Basic Message Display Utility

bmsg is invoked as follows:

```
bmsg [message_number [, message_number...]] [-v]
```

where:

message_number specifies the message to display.

-v requests bmsg to print the current version number, and exit. It overrides all other options.

If no *message_number* is given, bmsg prompts the user and displays the requested message. This continues until the user types Q to quit. An example bmsg operation follows:

```
% bmsg
Please enter message identifier (Q to quit):
ABCD0001
MESSAGE: ABCD0001
%s This is an example message string which starts with a string parameter.
PARAMETERS:
1. A string (%s)
CAUSE:
This field normally explains what condition(s) causes this message to be displayed.
EFFECT:
This field normally displays what effect the condition(s) may have had.
ACTION:
This field normally indicates what actions should be taken when this message is displayed.
Please enter message identifier (Q to quit):
Q
%
```

E.2 BMD: Sun Basic Message Database

BMD is a Unix directory. The BMD directory contains several message files, each representing one product component. Each product component owns a range of message numbers and stores message information in its own message file. The file is uniquely named *xxxx_msglib*, where *xxxx* is the product component identifier.

The BMD directory may be located anywhere on the user's search path, as specified by the environment variable \$PATH. If you want to modify a message file, copy it out of the BMD directory into the current working directory. If \$PATH includes the current directory “.”, the message file in the current working directory will be accessed rather than the version in the BMD directory.

E.3 Message Files

A message library is a text file containing multiple message entries. Each message has up to seven defining fields:

- Message number
- Message string
- Message string parameters
- Message display options
- Cause
- Effect
- Action

Message fields are delimited by blank lines. Messages are separated by a full line of hyphens ('-'). An example (fictitious) message entry follows:

```
MESSAGE:
ABCD0002   Line %s has reached the retry limit (%d)
PARAMETERS:
1. LINE name (%s)
2. Retry limit (%d)
CAUSE:
The specified LINE has attempted to retransmit a message retry
limit times.
EFFECT:
LINE moves into disconnected state.
ACTION:
1. Check the configured MAXDATA size for the line. If you are
   transmitting oversize frames they will be rejected.
2. Use the Operator to examine the line statistics. If the line
   condition is poor, it may be necessary to increase the retry
   limit. Otherwise, you may need to increase the retry timeout
   value.
```

Table E-1 describes the message string components.

Table E-1 Message String Components

Field	Function										
MESSAGE	The MESSAGE field contains the message number and its associated string. The message number, such as ABCD0001, is fixed. The message string is similar to a <code>printf</code> formatted string. The order of the parameters in the string is fixed. The message string can be up to 1016 characters long and cannot contain any blank lines. The following parameter insertion values are defined: <table><tr><td>%s</td><td>Inserts a string (<code>printf("%s")</code>).</td></tr><tr><td>%d</td><td>Inserts a decimal integer (<code>printf("%d")</code>).</td></tr><tr><td>%x</td><td>Inserts a hexadecimal integer (<code>printf("0x%08x")</code>).</td></tr><tr><td>%%</td><td>Escapes the '%' so this character can be included in string (<code>printf("%%")</code>).</td></tr></table>	%s	Inserts a string (<code>printf("%s")</code>).	%d	Inserts a decimal integer (<code>printf("%d")</code>).	%x	Inserts a hexadecimal integer (<code>printf("0x%08x")</code>).	%%	Escapes the '%' so this character can be included in string (<code>printf("%%")</code>).		
%s	Inserts a string (<code>printf("%s")</code>).										
%d	Inserts a decimal integer (<code>printf("%d")</code>).										
%x	Inserts a hexadecimal integer (<code>printf("0x%08x")</code>).										
%%	Escapes the '%' so this character can be included in string (<code>printf("%%")</code>).										
PARAMETERS	The parameters in the message string are described in the PARAMETERS field. Each parameter is defined on one line. This field can be up to 1024 characters and cannot contain any blank lines.										
OPTIONS	The OPTIONS field describes how the messages should be displayed. Options may include: <table><tr><td>NO_PRINT</td><td>Do not display this message.</td></tr><tr><td>NO_MESSAGE_NUMBER</td><td>Do not prefix the message with its number.</td></tr><tr><td>PROGRAM_NAME</td><td>Prefix the message by the program name.</td></tr><tr><td>PROCESS_ID</td><td>Prefix the message by the user process id.</td></tr><tr><td>TIME</td><td>Prefix the message by the date and time.</td></tr></table>	NO_PRINT	Do not display this message.	NO_MESSAGE_NUMBER	Do not prefix the message with its number.	PROGRAM_NAME	Prefix the message by the program name.	PROCESS_ID	Prefix the message by the user process id.	TIME	Prefix the message by the date and time.
NO_PRINT	Do not display this message.										
NO_MESSAGE_NUMBER	Do not prefix the message with its number.										
PROGRAM_NAME	Prefix the message by the program name.										
PROCESS_ID	Prefix the message by the user process id.										
TIME	Prefix the message by the date and time.										
CAUSE	The CAUSE field describes why the message is displayed. This field can be up to 1024 characters and cannot contain any blank lines.										
EFFECT	The EFFECT field describes the effect on the product component due to the condition described by the associated messages. This field can be up to 1024 characters and cannot contain any blank lines.										
ACTION	The ACTION field describes the actions the user or administrator should take when the associated message is displayed. This field can be up to 1024 characters and cannot contain any blank lines.										

DBCS



This appendix is for Pacific Rim users with double-byte character string (DBCS) functionality.

F.1 DBCS Keywords

The following keywords apply to sun3270x and pcft:

Enables sun3270x to process and display DBCS. Default is off.

```
-w (or -dbcs)
.dbcsMode: on|off
```

Selects the desired input method. Default is onspot if .dbcsMode is turned on.

```
-desiredImStyle
.desiredImStyle: onspot|offspot|root
```

Identifies the command to use for file transfer. Default is APVUFILE when .dbcsMode is turned on and IND\$FILE when it is off.

```
-pcftCommand
.pcftCommand: pcft_command
```

To display SO/SI characters and field characters on the sun3270x screen. Default is off.

```
-showBufferControls  
.showBufferControlsMode: on|off
```

Identifies a special dbcs<->euc table to use. Default is the SunLink table.

```
-W  
.dbcsEucTableFilename: dbcs_euc_table
```

Identifies a legacy dbcs<->euc table from SunLink 8 to use. Default is none.

```
-L  
.legacyDbcsEucTableFilename: legacy_dbcs_euc_table
```

Requests that sun3270x to display DBCS<->EUC translation table to /dev/stdout (or to filename specified by -O option) during initialization. Default is off.

```
-V  
.printDbcsEucTable: on|off
```

Requests that sun3270x to display the widget tree to /dev/stdout (or to filename specified by -O option) during initialization. Default is off.

```
-printWidgetTree:  
.printWidgetTree: on|off
```

Requests that sun3270x to display the font names in the font pool to /dev/stdout (or to filename specified by -O option) during initialization. Default is off.

```
-printFontPools:  
.printFontPools: on|off
```

Specify a font set for the 3270x.

```
-fontList:  
.fontList:  font_list
```

For example:

```
sun-song-medium-r-normal--16-140-75-75-c-140-gb2312.1980-0, \  
sun-song-medium-r-normal--16-140-75-75-c-70-iso8859-1  
  
- OR -  
  
*song*:      <":" must follow which ends the font list when only  
              a single font is specified. >
```

Specify the font to use as normal font.

```
-fontPoolNormal  
*fontPoolNormal:  normal_font
```

Specify the font to use as bold font.

```
-fontPoolBold  
*fontPoolBold:   bold_font
```



Specify the font to use in the OIA area.

```
-fontPoolOIA  
*fontPoolOIA:
```

Index

A

- Action menu, 3-27
- aliases
 - record, B-5
- alternate cursor, 1-7
- application tables, 9-7
- arguments
 - generic, A-8
- ASCII character sequence, B-6
- ASCII/EBCDIC translations, 4-9
- attention function, 3-37
- audience, xv

B

- backtab key, 3-35
- basic message database, E-2
- BMD, E-1
- bmsg display utility, E-2

C

- caps lock key, 3-34
- case or test key, 1-7
- CG3270 usage, 12-1
- characterization
 - key, A-28

- CICS, 6-2, C-1
- CICS options, C-7
- clear, 3-37
- CMS, C-1
- color display attributes, 3-6
- command_line, 7-4
- configuration, 1-7
 - display terminal
 - example, 8-4
 - SNA server
 - example, 8-4
- configuration file
 - SunLink PU2.1, 8-1
- configurations
 - coordinating, 2-3
- configuring SUNWopc1, 2-3
- conversational monitoring system, C-1
- cursor control, 3-34
- cursor movement
 - field to field, 3-35
- customizing keyboard, A-1

D

- data entry assist, 1-7
- data entry keys, 3-33
- DBCS

- file transfer, 6-1
- keywords, F-1
- defaults
 - light pen, 3-18
- defining
 - IBM key caps file, A-10
- delete function, 3-39
- dependencies
 - for file transfer, 6-1
 - starting sun3287x, 7-2
 - sunke, A-2
- Display dialog box, 4-5
- display interfaces, 1-1
- display terminals
 - IBM 3278, 1-4
- display window
 - color
 - functions, 3-7
- double cursor movement, 1-7
- DSC printers emulation, 7-1
- duplicating IBM key, 5-8
- dynamic update
 - SUNWopc1 resource, 4-4

E

- EBCDIC characters, 1-5
- Edit menu, 3-25
- editing functions, 3-38
- EHLLAPI server
 - starting, 6-2
- EHLLAPI session name, 3-5
- emulation
 - IBM 3278, 3-1
- emulation screen for 3270
 - status line, 3-28
- Enter key function, 3-37
- error message
 - handling scheme, E-1
- error messages, E-1
- Escape key sequence, 13-1
- exiting
 - keyboard map display, 5-9

F

- field control, 3-39
 - clear field, 3-39
 - delete left, 3-39
 - DUP, 3-39
 - erase input, 3-39
 - erase to end-of-field, 3-40
- file format
 - keyboard map, B-2
- File menu, 3-25
- File transfer
 - accessing, 3-27
- file transfer, 6-1
 - menu defaults, 3-15
 - starting, 6-2
- file transfer options, 6-4
- file transfer status, 6-4
- function keys
 - PF1 to PF24, 3-37
 - user-defined, 3-44
- functions
 - miscellaneous, 3-11

G

- GDDM commands, 12-1
- graphical interfaces
 - support, 3-1
- Graphics, host
 - sun3270x, 3-19
 - suntn3270x, 11-4

H

- help
 - on-line, 10-2
- Help key sequence, 13-1
- home key, 3-36
- horizontal_format, 7-4
- Host graphics keywords
 - sun3270x, 3-19
 - suntn3270x, 11-4
- host_name, 7-3

I

- IBM key
 - moving, 5-6
- IBM key caps file
 - sample, A-13
- ibmCaps
 - record, B-3
- insert mode, 3-38
- installation scripts
 - provided, 2-1
- installing SUNWopc1, 2-2

K

- keyboard functions, 3-31
- keyboard geometry file, A-19
 - sample, A-20
- keyboard input
 - 3270, 3-30
- keyboard layout, 5-6
 - definition files, A-3
- Keyboard map
 - accessing, 4-5
- keyboard map
 - displaying, 5-1
 - layout, 5-2
 - updating, 5-6
- keyboard map utility, B-1
- keyboard mapper
 - file, 5-4
 - menu bar, 5-4
 - next, 5-4
 - other info, 5-4
 - quit, 5-4
 - title information, 5-4
- keyboard mapping, 1-7
- keyclick, 1-7
- Keywords
 - suntn3287, 7-7
- keywords, 3-3, A-3
 - ehllapi_name*, 13-4
 - host_name*, 13-4
 - key_map_pathname*, 13-4

- LU attachment, 3-3
 - lu_name*, 13-4
 - port_number*, 13-4
 - pu_name*, 13-5
 - trace_flag*, 13-5

- Keywords, sun3270x
 - DBCS, F-1
 - host graphics, 3-19
- Keywords, suntn3270x
 - host graphics, 11-4
 - miscellaneous, 11-3

L

- light pen, 3-18
- logical data scope
 - tracing, 10-3
- LOGMODE tables, 9-5
- LU directive, 8-2

M

- main display
 - window attributes, 3-9
- map attributes
 - keyboard, 3-4
- mapping ASCII characters, 1-5
- mapping IBM key caps, A-31
- message files, E-3
- message library
 - updates, 4-8
- mode button, 5-5
- Motif, 11-1
- Mouse dialog box, 4-6

N

- NCP/VTAM GEN, 9-2
- new keyboard layout
 - creating, A-9
- new line key, 3-36

O

- on-line help, 10-2
- operator status information, 10-2
- Options
 - sun3270x menu, 3-25, 3-27
- organization
 - manual, xvi

P

- path_name, 7-4
- PC file transfer, C-1
- pcft utility, C-1
- port_number, 7-3
- preface, xv
- printer
 - IBM 3287, 1-4
- program access keys, 3-37
- pu_name, 7-3

R

- READY prompt, C-1
- receive (CICS), C-9
- receive (CICS) example, C-10
- receive (TSO), C-6, C-7
- receive (VM/CMS), C-12
 - examples, C-13
- recovery steps, 10-4
- removing IBM key, 5-9
- reset key, 3-38
- running sun3270x, 3-1
- running SUNWopcl, 2-1

S

- sample configurations, 2-1
- saving
 - keyboard map updates, 5-9
- Screen Color Mode, updating, 4-6
- Screen Dump, 4-7
- screen format, 1-5

- screen formats
 - 3270, 3-30
- SCS control codes, 7-1
- SCS emulation, 7-1
- send (CICS), C-8
 - examples, C-9
- send (VM/CMS), C-11
 - examples, C-12
- send TSO, C-3
- send TSO examples, C-5
- send TSO syntax, C-4
- server configuration
 - SunLink PU2.1 SNA, 8-1
- Settings menu, 3-27
- Settings, brx3270 display, 4-6
- shell prompts, xx
- shift key, 3-34
- single key layout, 5-5
- SNA BIND request, 7-1
- SNA configuration, 9-1
- SNA resources, 1-3
- special functions, 3-40
 - copy, 3-43
 - cursor select, 3-41
 - cut, 3-43
 - cut, copy and paste, 3-41
 - paste, 3-43
 - print, 3-41
 - selector light pen, 3-41
 - system request, 3-40
- start/stop data entry, 3-36
- starting sun3287, 7-2
- starting sunpu2.1, 2-4
- starting SUNWopcl, 2-5
- startingsun3270x, 3-2
- Status line
 - sun3270x, 3-28
- status line
 - components, 3-29
- Stopping
 - sun3270x, 3-28
- stopping sunpu2.1, 2-6

- stopping suntn3270x, 11-6
- stream_command, 7-4
- sun3270 sample map file, B-7
- sun3270 trace, D-2
- sun3270 tracing capabilities, D-1
- sun3270tty
 - keyboard map utility, B-1
- sun3270tty dependencies, 13-2
- sun3270tty examples, 13-6
- sun3270tty intensified support, 13-3
- sun3270tty keywords, 13-3
- sun3270tty status line, 13-2
- sun3270x
 - status line, 3-28
 - stopping, 3-28
- sun3270x resources, 4-1
- sun3270x window, 3-23
- sun3270x window layout, 3-23
- sun3270x window parts, 3-24
- sunke
 - keyboard functions
 - teaching, A-28
 - starting, A-1
- sunke customizing, A-8
- sunke files, A-2
- sunop, 2-6
- suntn3270 dependencies, 11-2
- suntn3270 examples, 11-7
- suntn3270 keywords, 11-2
- suntn3270x, 11-1
 - status line, 11-6
- suntn3287
 - keywords, 7-7
 - starting, 7-7
- SUNWopcl applications, 1-3
- SUNWopcl capabilities, 1-1
- SUNWopcl client, 1-3
- SUNWopcl emulation, 1-1, 1-4
- SUNWopcl resource example, 4-3
- SUNWopcl screen, 1-5

T

- tab forward, 3-36
- Telnet 3270 emulation, 11-1
- terminal map record, B-5
- Time Sharing Option, C-1
- trace points, D-1
 - format, D-1
- trace_flag, 7-6
- transferring files, 6-2
- translation table
 - sample
 - ASCII/EBCDIC, 4-10
- translation_name, 7-6
- troubleshooting
 - common problems, 10-5
- troubleshooting sun3270, 10-1
- troubleshooting utilities
 - BMD, 10-1
 - logical data scope, 10-1
 - SunLink PU2.1 operator, 10-1
- TSO, 6-2
- TSO options, C-3
- TSO READY, 6-2

U

- unmapped IBM keys, 5-5
- unsupported
 - terminal functions, 1-7
- User Keys dialog box, 4-7
- using sun3287, 7-1

V

- vertical_format, 7-5
- VM/CMS, 6-2
- VM/CMS options, C-10

W

- window attributes, A-4
- window colors, A-6

X

X keywords, 3-3, 3-10

X Window Manager, 11-1

X Window standard, 11-1

Xserver, 3-12

X-terminals, 11-1